In compliance with the
Canadian Privacy Legislation
some supporting forms
may have been removed from
this dissertation.

While these forms may be included
in the document page count,
their removal does not represent
any loss of content from the dissertation.

University of Alberta

A Probability Model to Refine Word Alignments

by

Colin Andrew Cherry

A thesis submitted to the Faculty of Graduate Studies and Research in partial fulfillment of the requirements for the degree of **Master of Science**.

Department of Computing Science

Edmonton, Alberta
Fall 2003

National Library
of Canada

Acquisitions and
Bibliographic Services

395 Wellington Street
Ottawa ON K1A 0N4
Canada

Bibliothèque nationale
du Canada

Acquisisitons et
services bibliographiques

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence
ISBN: 0-612-87738-8
Our file Notre référence
ISBN: 0-612-87738-8

The author has granted a non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of this thesis in microform, paper or electronic formats.

The author retains ownership of the copyright in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de cette thèse sous la forme de microfiche/film, de reproduction sur papier ou sur format électronique.

L'auteur conserve la propriété du droit d'auteur qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou aturement reproduits sans son autorisation.

Canada

University of Alberta

Library Release Form

Name of Author: Colin Andrew Cherry

Title of Thesis: A Probability Model to Refine Word Alignments

Degree: Master of Science

Year this Degree Granted: 2003

Permission is hereby granted to the University of Alberta Library to reproduce single copies of this thesis and to lend or sell such copies for private, scholarly or scientific research purposes only.

The author reserves all other publication and other rights in association with the copyright in the thesis, and except as herein before provided, neither the thesis nor any substantial portion thereof may be printed or otherwise reproduced in any material form whatever without the author's prior written permission.

Date: Aug 15, 2003

University of Alberta

Faculty of Graduate Studies and Research

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies and Research for acceptance, a thesis entitled **A Probability Model to Refine Word Alignments** submitted by Colin Andrew Cherry in partial fulfillment of the requirements for the degree of **Master of Science**.

_____

_____

_____

Date: July 21, 2003

Abstract

Parallel texts have recently been recognized as a valuable resource for machine translation researchers. These texts, which present the same information in two or more languages, are a rich source of translation knowledge. However, the texts themselves are of limited use without a guide to indicate what portions of the two languages correspond to one another.

The goal of word alignment algorithms is to link words in bilingual sentence pairs, in order to indicate which words are translations of one another. We present a statistical model for computing the probability of an alignment given a sentence pair. This model allows easy integration of context-specific features. A system called ProAlign has been built around this model to refine an existing word alignment. We show that ProAlign is competitive with the current state of the art in word alignment, and that its central probability model is better suited to alignment refinement than alternative models.

Acknowledgements

First of all, I would like to thank all those I care about back East, but especially Mum, Dad and Becky, for being supportive (and not that surprised) when I decided to abandon the corporate world for a graduate degree out West. I would also like to thank NSERC and ICORE for making the life of a graduate student so attractive with their funding.

This research would not have been possible if it were not for the support and patience of my fantastic supervisor, Dekang Lin. I would also like to thank the University of Alberta Machine Learning Reading Group and the Natural Language Group for providing me with a forum to present my ideas and receive feedback. Among the members of those groups, I would like to especially thank Russ Greiner for the advice early on, and Dan Lizotte for the sympathetic ear.

I would like to thank Franz Och for providing us with manually aligned evaluation data. The work presented in this thesis is funded by and jointly undertaken with Sun Microsystems, Inc. I would like to thank Finola Brady, Bob Kuhns and Michael McHugh for their help. Finally, I would like to thank Ted Pederson and Rada Mihalcea for arranging this year's workshop on working with parallel texts, and providing us with a forum to test our algorithms.

Table of Contents

1 Introduction **1**
- 1.1 Motivation . . . 2
- 1.2 Objectives . . . 3
- 1.3 Outline . . . 4

2 Related Work **6**
- 2.1 Sentence Alignment . . . 6
- 2.2 Candide Statistical Machine Translation . . . 7
  - 2.2.1 General Candide Translation Model . . . 8
  - 2.2.2 Candide Model 1 . . . 10
  - 2.2.3 Candide Model 2 . . . 15
  - 2.2.4 HMM Alignment . . . 18
  - 2.2.5 Candide Model 3 . . . 21
  - 2.2.6 Candide Models 4 and 5 . . . 27
  - 2.2.7 GIZA++ and Model-Independent Extensions . . . 29
- 2.3 Word-to-word Models of Translational Equivalence . . . 31
  - 2.3.1 Probabilistic Model . . . 35
  - 2.3.2 Explicit Noise Model . . . 38
  - 2.3.3 Word-to-word Model Summary . . . 39
- 2.4 Maximum Entropy . . . 40
  - 2.4.1 Features in Maximum Entropy . . . 41
  - 2.4.2 Modeling the Distribution . . . 43
  - 2.4.3 Maximum Entropy to Enhance Candide . . . 44
- 2.5 Syntax in Translation . . . 45

3 Probability Model **48**
- 3.1 Derivation . . . 48
- 3.2 Parameter Estimation . . . 51
- 3.3 An Illustrative Example . . . 53

4 Alignment Algorithm **57**
- 4.1 High-level Description . . . 57
- 4.2 Probability Evaluation . . . 58
  - 4.2.1 Features . . . 59
  - 4.2.2 Smoothing . . . 60
- 4.3 Search . . . 62
  - 4.3.1 Constraints . . . 63
- 4.4 Training . . . 66
  - 4.4.1 Initial Alignment . . . 66
  - 4.4.2 Sampling . . . 67

5 Experimental Design and Results **68**
- 5.1 Evaluation Procedure . . . 68
- 5.2 Comparisons to the State of the Art . . . 70
  - 5.2.1 In-house Experiments . . . 70
  - 5.2.2 Word Alignment Shared Task . . . 71

5.3 Contributions of Algorithmic Components 74
5.4 Validating the Probability Model 75

6 Future Work **77**
6.1 Soft Constraints 77
6.2 Maximum Entropy 78
6.3 Synchronous Parsing 79

7 Conclusion **81**

Bibliography **82**

List of Tables

2.1 A Co-occurrence Contingency Table 33
2.2 An Example of Maximum Entropy Feature Templates 45

3.1 Example Probability Tables 54

4.1 Context Ratio Smoothing Example 62

5.1 Comparison with GIZA++ Candide 71
5.2 Shared Task Results: French-English 72
5.3 Shared Task Results: Romanian-English 73
5.4 Evaluation of Components 74
5.5 $P(A|E,F)$ Model versus Candide Model 1 76

List of Figures

2.1 An Example Alignment 9
2.2 An Example Candide Model 1 Alignment 15
2.3 An Example Candide Model 2 Alignment 17
2.4 An HMM Alignment in Grid Format 19
2.5 An Example Candide Model 3 Alignment 26
2.6 An Example of Refined Alignment Combination 31
2.7 An Illustration of an Indirect Association 34
2.8 A Sentence Parsed Using a Dependency Grammar 46

3.1 An Example Aligned Corpus 54
3.2 An Alternative Alignment: A' 55

4.1 Feature Extraction Example 60
4.2 A Cohesion Constraint Violation 64
4.3 An Aligned Sentence Pair with English Dependency Tree 65

Chapter 1

Introduction

Machine translation is the study of computational tools built to aid the translation of human languages. The holy grail of machine translation is a system that will take text from one language and translate it automatically into another. Until relatively recently, the most effective systems in automatic machine translation have relied on rule-based methods, with manually constructed translation rules and dictionaries. These systems are effective for many tasks where a rough translation is good enough. Unfortunately, the quality of translations produced by these hand crafted systems has reached a sort of plateau [16], as creating the necessary linguistic knowledge manually is both difficult and expensive.

A potential source of translation knowledge is **parallel text**. When used in the machine translation community, the term parallel text refers to two or more texts that are mutual translations of one another. The terms **parallel corpus** and **bitext** also refer to parallel texts, with bitext referring to the special case where there are only two languages involved. By far the most famous parallel text is the Rosetta Stone. This artifact recorded the honors awarded King Ptolemy V in three different ancient languages [40]. Once discovered, the stone provided historians with a starting point, which allowed them to decipher the ancient Egyptian hieroglyphics.

Today, large parallel corpora are beginning to emerge: bilingual governments record the proceedings of their meetings in all official languages; corporations publish technical and promotional materials in multiple languages to reach a larger market; and web pages for multinational organizations are often multilingual to satisfy all of their members. With the rise of the CD-ROM and the Internet, many of these resources are now available to researchers in an electronic format, which has drawn the attention of the machine translation community. There is a wealth of translation knowledge captured in these documents, but transferring this information to computers remains a challenge.

It is not clear, at first glance, how a computer can make use of parallel text to gain translation knowledge. After all, the parallel text provides a translation for an entire document, not instructions on how to translate documents in general. Fortunately, it appears that having a computer reverse engineer the translation process is a less challenging goal than creating a translation from scratch. Two major enabling technologies have emerged from the natural language processing community in order to aid in the extraction of knowledge from bitext, the most common type of parallel text. These technologies are referred to as **sentence alignment** and **word alignment**. Both tasks are able to be accomplished in an unsupervised manner; the only training data required is the bitext itself. Sentence alignment is the study of automatically determining which sentences in parallel text are translations of one another. This problem has proved quite tractable, as sentences in a paragraph tend to stay in roughly the same order after translation. Sentence alignment is considered to be a solved problem, with the most advanced methods achieving 98.5% accuracy on reasonable parallel texts [40]. With the ability to reliably tell which sentences correspond to one another, the next logical step is to determine how elements of the sentences correspond to one another. The most common method of doing so is word alignment, which automatically determines which words in a sentence pair are translations of one another. As words within a sentence can move around a fair amount during translation, the problem of word alignment has proved to be quite difficult. It remains a challenging and active area of research. Word alignment will be the focus of this thesis.

1.1 Motivation

Once text has been aligned word-by-word it opens up a world of possibilities for other learning algorithms. In this way, word alignment is an enabling technology, providing a resource that other technologies can draw upon. This section will list a few of the possible uses for word-aligned bitexts, including some obvious uses, such as dictionary construction and automatic translation, and some that are not so obvious, such as translation verification and word sense disambiguation.

Given a word alignment, bilingual dictionaries can be constructed automatically by examining how often pairs of words are linked in the aligned bitext [28]. This technology is in high demand. Our team at the University of Alberta has been commissioned by Sun Microsystems to construct a English-French dictionary of product-specific terms from Sun manuals. Similarly, a group at the Canadian National Research Council has used bitext

to construct an English-Inuktitut dictionary [27] from the debates and proceedings of the Nunavut Legislative Assembly. This dictionary can then used to enforce consistency, as some of the Inuktitut terms are being written down for the first time when they are recorded in the minutes.

Word alignments are needed to provide data for some types of automatic machine translation system. In some cases, the parameters used in creating word alignments can be used directly in translation [4]. In others cases, an existing alignment can be used to assign statistics for phrase-based translation [18], or to enhance the process of using entire sentence pairs in example-based translation [26]. Finally, one can automatically construct the sort of translation rules that are traditionally constructed by hand by mining the data contained in word-aligned bitexts [5].

Another potential application of word alignment technology is automatic verification of human-created translations. Programs such as the University of Montreal's TransCheck system [23] use sentence-level alignments and hand constructed "anti-lexicons" to check a translation automatically, like a bilingually aware spelling and grammar checker. One could imagine extending such a technology to make use of a word alignment process in order to monitor quality and consistency of translation. Sentences with words that do not align properly could be flagged to be reviewed by a human translator.

Finally, there exist applications of word alignment that have very little to do with translation. One can think of the different ways of translating a word as indicators of the different senses of the word. For example, when dealing with the word *sentence*, one can tell the judicial sense from the syntactic sense in an aligned French-English bitext by seeing whether or not *sentence* was translated using *peine* or *phrase*. Using this intuition, an aligned bitext can be used to create large sets of monolingual examples of word sense usage. These examples can then be used to train word sense disambiguation systems [13].

1.2 Objectives

The work presented in this thesis is a novel alignment system capable of producing high quality word alignments from bitext. As this system will be the first component of a number of different systems, we will focus on creating high precision word alignments, meaning that we seek an algorithm that links words only when there is a high degree of certainty that the link is correct. This will prevent excess noise from affecting the technologies that will use the word alignments. The entire system will be presented as a method for refining an

existing alignment, meaning that it can start with a noisy word alignment, and iteratively improve it. Current state-of-the art alignment systems are not well-suited to this particular task.

As a part of this larger and more visible goal, we propose a new statistical model designed specifically for capturing the information present in a noisy word alignment. Given a bilingual sentence pair, this model assigns probability values to various alignments. To allow us to continue to test ideas about word alignment, this model should be easy to understand, and easy to extend. To achieve this goal, we incorporate modular, context-related features that can be added to and removed from the model with relative ease.

The experiments conducted in this thesis will be designed to test the quality of this system's alignments, and the utility of the probability model in achieving its goal of alignment refinement. We will show that this system is competitive with the current state of the art, and that the model presented here is better suited to alignment refinement than other, established probability models.

1.3 Outline

This section will briefly describe the chapters that will follow in this thesis. Chapter 2, Related Work, will outline the previous work that has been most relevant to the research presented here. Special attention will be payed to competing and influential translation models, such as the Candide system and Melamed's models of translation equivalence. Other systems which provided some degree of inspiration will be described more briefly.

The following two chapters will describe the novel work created as a result of this research. Chapter 3, Probability Model, will present the probability model that is the centerpiece of our alignment algorithm. The model will be motivated and derived, while parameter estimation and alignment evaluation will be described through a number of small examples. Chapter 4, Alignment Algorithm, will describe the search-based word alignment algorithm that has been built around this probability model. This chapter will be broken up into sections according to algorithmic components, with the components being explained in the order in which they are most easily understood.

Chapter 5, Experimental Design and Results, will present the evaluation framework used by the machine translation community to validate word alignment algorithms. This framework will then be used to test the alignment algorithm against the state of the art, evaluate individual algorithmic components, and evaluate the utility of the probability model

itself. Chapter 6, Future Work, will outline ways in which this research could be extended to improve word alignment quality. Chapter 7, Conclusion, will summarize the results presented in this thesis.

Chapter 2

Related Work

This chapter will outline the work that has inspired and influenced the development of the models and algorithms described in this thesis. These works fall into three main categories: enabling technologies (including sentence alignment), previous alignment models (including the Candide translation models and Melamed's models of translational equivalence), and alignment enhancements (including maximum entropy and syntactic methods). This chapter will focus on briefly explaining the work done in these areas to the reader, so that the methods developed in this thesis can be understood in the context of previous work.

2.1 Sentence Alignment

This thesis deals with the automatic alignment of words within sentences. The alignment method defined here and others like it are made possible by an established technology that allows large corpora to be sentence aligned. In this way, pairs of sentences that are mutual translations of one another can be fed to the alignment algorithm, which works on a sentence-by-sentence basis.

Most methods for sentence alignment, like the one defined in [15], take advantage of the fact that sentences are generally transcribed in roughly the same order during translation. Sentences also tend to maintain the same relative lengths between languages. When given two sentence-delimited parallel texts, [15] looks at sentences in each text in terms of their lengths in characters. It uses a simple probability model together with dynamic programming in order to determine an optimal arrangement of sentences, assuming that sentences can be translated normally, merged, inserted, or deleted during translation. Each of these operations has an empirically derived probability of occurring. This method is generally considered accurate enough for most sentence alignment tasks. Furthermore, this method is aware of when it is not certain. Therefore, a paragraph whose sentence alignment is uncer-

tain can simply be left out of the bitext, rather than introduce extra noise into the input for word alignment. More complex bitext correspondence methods exist for added accuracy. Known as bitext maps [30], these methods take notions like cognates and known translations into account along with sentence boundaries in order to create a rough map of points of correspondence in the parallel corpus. The bitext map generator uses pattern matching technology along with geometric intuitions to tell true points of correspondence from false ones. Armed with bitext map, one can do sentence alignment, or read word co-occurrence counts directly from the map.

Sentence alignment is considered a solved problem. The relatively simple method described in [15] is capable of aligning 80% of the sentences in a corpus with an error rate of less than 1%. The other 20% of the sentences can be discarded to avoid polluting the training set. All of the alignment algorithms described in this thesis will take a sentence aligned corpus like the one outputted by [15] as input.

2.2 Candide Statistical Machine Translation

The Candide (or IBM [4]) statistical machine translation models have been extremely influential in computational linguistics in the last decade. These models are generally credited with demonstrating how much one can accomplish in language using only statistics, without built-in linguistic knowledge. This work is also responsible for the creation of the field of statistical machine translation (SMT) itself, and extensions of this work are considered to be the current state of the art in this field. Thus, any discussion on statistical word alignment should begin with a description of this system.

The Candide group envisioned translation as a process that can be modeled statistically as a noisy channel decoding problem. The task of translation became a search for the target sentence E that is most likely given the source sentence F, or $\text{argmax}_E P(E|F)$. To facilitate the modeling process, $P(E|F)$ is decomposed as $P(E) \cdot P(F|E)$, which is proportional to $P(E|F)$ when F is held constant. This decomposition allows a monolingual language model [25] $P(E)$ to provide input to the translation process. The inclusion of the language model means that the translation model $P(F|E)$ will receive help when evaluating some potentially difficult aspects of translation, such as word order in the target language [17]. This decomposition breaks translation into three problems: language modeling ($P(E)$), translation modeling ($P(F|E)$) and decoding (the search for $\text{argmax}_E P(E)P(F|E)$). Among these three tasks, only translation modeling is related to word alignment. The remainder of

this section will discuss the details of modeling and training $P(F|E)$.

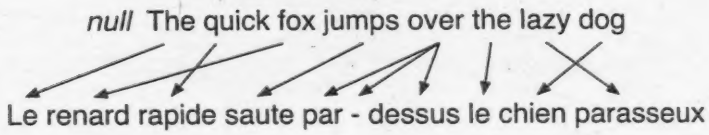
2.2.1 General Candide Translation Model

According to the noisy channel model, the role of the translation model $P(F|E)$ is to characterize how sentences in the target language become sentences in the source language. When translating a source sentence, the decoding process must use the translation model and the language model to recover the original target sentence that created the source sentence. Moving backward in this manner can create some degree of source-target confusion, as the target generates the source. To avoid confusion, the source language will be referred to as French and the target language as English for the remainder of this review of the Candide system. The translation model, $P(F|E)$ is designed from the perspective of an English sentence, E generating a French sentence, F.

To make computation of the translation model tractable, one needs some sort of generative story that decomposes the creation of F from E into a series of events. Candide models the English to French transformation on a word to word basis. Each word in the French sentence is generated by a single word in the English sentence; conversely, each word in the English sentence generates zero or more French words. In order to describe those cases where a French word occurs with no corresponding word in the English sentence, there is an additional English word located at position zero called the *null* word. This word is responsible for all spurious words, such as the French word *de*, which is often included between noun-noun compounds with no matching word in English.

Let m be the length in words of the French sentence and let n be the length of the English sentence. Each of the m French words has $n + 1$ choices for its generator: the n English words and *null*. Therefore, there are $(n + 1)^m$ possible ways to generate F from E. Each possible generation story for a sentence pair can be characterized by an alignment, which is an annotated version of the French sentence. Formally, an alignment A is an ordered sequence of m integers $a_1 \ldots a_m$ that indicate which English word generated each of the m French words. For the purposes of this thesis, a graphical alignment format will be used to describe alignments, although a text-based format can be used just as easily. An example of both formats is shown in Figure 2.1. In this case, most English words have generated only one French word, though *over* generated three words, while *null* did not generate any words. It is important to note that this generation-based definition of alignment is specific to the Candide models of SMT. More general definitions of word alignment exist, and will be described later in this thesis.

Graphical format:



Text Format:

$null_0$ The_1 $quick_2$ fox_3 $jumps_4$ $over_5$ the_6 $lazy_7$ dog_8

Le{1} renard{3} rapide{2} saute{4} par{5} -{5} dessus{5} le{6} chien{8} paresseux{7}

Figure 2.1: An Example Alignment

For a given sentence pair, all possible alignments should be taken into account when calculating $P(F|E)$. An intermediate term, $P(F, A|E)$, is defined as the probability of generating F from E according to the alignment A. Given this term, and the set of all possible alignments $\mathcal{A}$, we can calculate $P(F|E)$ with the following equation:

$$P(F|E) = \sum_{A \in \mathcal{A}} P(F, A|E) \tag{2.1}$$

As the focus of this thesis is word alignment and not translation, our primary concern in examining the Candide system will be calculating the most likely, or Viterbi alignment for a sentence pair: $\text{argmax}_{A \in \mathcal{A}} P(A|E, F)$. Candide-based alignment systems can calculate the likelihood of an alignment for a sentence pair with the following manipulation:

$$P(A|E, F) = \frac{P(F, A|E)}{P(F|E)} \tag{2.2}$$

and therefore:

$$\text{argmax}_{A \in \mathcal{A}} P(A|E, F) = \text{argmax}_{A \in \mathcal{A}} \frac{P(F, A|E)}{P(F|E)} = \text{argmax}_{A \in \mathcal{A}} P(F, A|E)$$

At this point, one can see that the two objectives of translation modeling and word alignment are tied. As the discussion of the training process continues, we will see that under the Candide framework, it not possible to have one without the other. To find word alignments, all that is left to do is to define a tractable parameterization of $P(F, A|E)$, and to then determine an unsupervised method to set those parameters.

Fortunately, the definition of A has supplied a starting point one can use to parameterize $P(F, A|E)$. Each word in F is generated by a word in E; by fleshing out this generation process, one can create a parameterized equation. The process begins with only the English sentence existing. First, the process selects a length for the French sentence according to the

probability distribution $P(m|E)$. Next, for each French position j (where $1 \leq j \leq m$), the process selects an English sentence position a_j (where $0 \leq a_j \leq n$) to generate the French position. It then selects an appropriate French word f_j to place at that position. At every step, an ideal process takes all aspects of the English sentence and all decisions made up to that point regarding the French sentence and the alignment into account. This produces the following equation:

$$P(F, A|E) = P(m|E) \cdot \prod_{j=1}^{m} P(a_j|a_1^{j-1}, f_1^{j-1}, m, E) P(f_j|a_1^j, f_1^{j-1}, m, E) \qquad (2.3)$$

This equation makes no simplifying assumptions regarding any of its distributions. At each step in the process, all relevant information is taken into account. The equation is also quite intractable, especially since it conditions each of its component distributions on the entire English sentence E, which in most interesting cases has not been seen before. It is important to note that Equation 2.3 is only one of many possible ways to decompose $P(F, A|E)$, other decompositions will be introduced in some of the more complex Candide models.

The Candide system approaches the problem of intractability with several separate models, where the first two models are specific simplifications of Equation 2.3. The idea is to start by training models that are easy to work with, but also make many simplifying assumptions. The system then boot-straps to more complicated models that make less assumptions, but are not as well-behaved computationally. At each stage, the parameters from an earlier stage will be used to start training the new model at a point in parameter space that is close to its goal. Each model will be presented in the following subsections, starting with the simplest model and moving toward the most complicated. Each model will be described in terms of the simplifications it makes to its base "true" model, either Equation 2.3 or another more complex decomposition of $P(F, A|E)$. Each model has its own training process, which will also be described. For complete details, please see [4]. The following sections will briefly describe the five original Candide models, as well as an HMM model, which has been used successfully as a component of the Candide system in [37].

2.2.2 Candide Model 1

Before discussing the first model in the Candide hierarchy, we must introduce some essential terminology. In natural language processing, there are two meanings for the term *word*. To avoid confusion, we avoid use of *word* altogether, and instead refer to **tokens** and **types**. A token is a specific occurrence of a word in text. All tokens that appear to be identical to a

natural language system have the same type. In word alignment, we link individual tokens, not types.

The simplest and most well-behaved Candide model is Candide Model 1. This model makes several simplifying assumptions that essentially make the contribution of each French token to $P(F, A|E)$ independent of all factors except for the token's word type and the type of its generating English word. To do this, the following terms in Equation 2.3 are relaxed:

1. $P(m|E)$ is assumed to be independent of both m and E. This term becomes a constant ϵ.

2. $P(a_j|a_1^{j-1}, f_1^{j-1}, m, E)$, the position selector, is assumed to depend only on the length of the English sentence, n. It becomes $\frac{1}{n+1}$, to make all positions in E equally likely.

3. $P(f_j|a_1^j, f_1^{j-1}, m, E)$, the French type selector, is assumed to depend only on the French type and the type of the English generator indicated by a_j. It becomes $P_t(f_j|e_{a_j})$.

The final equation for Candide Model 1 probability is:

$$P(F, A|E) = \frac{\epsilon}{(n+1)^m} \prod_{j=1}^{m} P_t(f_j|e_{a_j}) \tag{2.4}$$

These new parameters have meanings that are far more intuitive than their counterparts in Equation 2.3. ϵ is simply a small, constant value. It represents the assumption that all French sentence lengths are equally likely, regardless of the length of the English sentence. One can think of it as one divided by the size of the longest French sentence expected. The position selector is determined completely from the two current sentences, and requires no training. That leaves only the French type selector, $P_t(f_j|e_{a_j})$. This term is most easily thought of as being proportional to how often e_{a_j} generates a word with type f_j, compared to how often it generates any other word type. Since it is a probability distribution, we are guaranteed that for a French vocabulary V and a constant English type e:

$$\sum_{\forall f \in V} P_t(f|e) = 1$$

Therefore, the strength of a connection between an English word e and French word f cannot be increased without weakening a connection between e and f' where $f' \neq f$. Whether or not this notion is linguistically intuitive is debatable; however, this constraint on parameter values will prove to be invaluable in guiding the training process.

$P_t(f|e)$ values will be supplied by a look-up table learned from a sentence-aligned parallel corpus. We will use the notation P_x to differentiate between the various probability tables that will be introduced in later models, where x indicates the table being used. In this case, t stands for the translation table. The notion of a translation table that stores $P_t(f|e)$ values for French-English word pairs will be the one constant factor that unites all of the models involved in the Candide system. This table will be created using the Expectation Maximization, or EM algorithm [10]. EM estimates a probability distribution over some hidden variable, given observable data related to the variable. The hidden variable is often some sort of intermediate step in the generation of the observable data. The EM algorithm begins with a random or flat distribution describing the behavior of the hidden variable. It collects statistics on the hidden data from observable data, based on its current distribution. The collected statistics are then used to update its current hidden variable distribution, and the process repeats. This algorithm is guaranteed to produce a distribution that increases the maximum likelihood of the data, until it reaches a local maximum in parameter space. The specifics of the algorithm as a general estimation procedure are outside of the scope of this thesis, but EM applied to Candide models will be described in depth in the following sections.

In the Candide models, the observable data is an English-French sentence pair; the hidden data is the alignment that describes the creation of the French sentence from the English. In Model 1 specifically, the parameters to be manipulated are the $P_t(f|e)$ values. The Candide EM process defines the likelihood of the data to be $P(F|E)$; therefore, as the algorithm moves through parameter space, the value of $P(F|E)$ for the corpus will increase with each iteration until the parameters reach some stationary point. The following paragraphs will detail the algorithm produced by applying EM to Candide Model 1. For a complete description of the mathematical motivation behind this process, please see [4].

Model 1 begins with a flat translation table. That is, if v is the size of the French vocabulary V, for all (e, f) word pairs $P_t(f|e) = 1/v$. Using Equation 2.4 and the current parameter values, one can calculate $P(F, A|E)$ for every possible alignment for each sentence pair. With Equation 2.2, these alignment scores can be converted to $P(A|E, F)$ values. Initially, all alignments will be equally likely, but remember that the algorithm will return to this point with new parameter values on the next iteration. Given these values, the algorithm can collect fractional link counts from each sentence. We say that f and e were

linked $c(f|e; F, E)$ times in the sentence pair (E, F), where:

$$c(f|e; F, E) = \sum_{A \in \mathcal{A}} P(A|E, F) \cdot link(e, f; E, F, A) \tag{2.5}$$

In this equation, $link(e, f; E, F, A)$ is the number times words with types e and f are connected by the alignment A. Mathematically, the link count for two words in a sentence pair, given an alignment A is:

$$link(e, f; E, F, A) = \sum_{j=1}^{m} \delta(f, f_j)\delta(e, e_{a_j}) \tag{2.6}$$

where δ is the Kronecker delta function. This function is equal to 1 only when its two arguments are identical, otherwise it is equal to 0. In our sample alignment in Figure 2.1, the types *the* and *le* are linked two times.

Given these counts, Candide can then update its translation tables. $P_t(f|e)$ is the probability that e generates f and not some other French word. Given a corpus of s sentence pairs, (E_1, F_1), (E_2, F_2), ... , (E_s, F_s), and the current translation model, the maximum likelihood estimator for $P_t(f|e)$ is the count for the pair (e, f), normalized so that the probability of all generations involving e sum to 1:

$$P_t(f|e) = \frac{\sum_{i=1}^{s} c(f|e; F_i, E_i)}{\sum_{f' \in V} \sum_{i=1}^{s} c(f'|e; F_i, E_i)} \tag{2.7}$$

With the $c(f|e; F, E)$ values from Equation 2.5 in hand, one can calculate a new table of $P_t(f|e)$ values. However, there are $(n+1)^m$ possible alignments for every sentence. For reasonably sized sentences, this can easily become far too many alignments to enumerate. Fortunately, the assumptions made by Model 1 allow one to get the necessary counts from all possible alignments without explicitly enumerating them. This stems from the fact that, under Model 1's simple assumptions, the selection of a French word f_j's generator has no bearing on the selection of $f_{j'}$'s generator, for any $j' \neq j$. Therefore, a new counting equation that looks at the likelihood of each f's generator independently can be devised:

$$c(f|e; F, E) = \frac{P_t(f|e)}{\sum_{i=0}^{n} P_t(f|e_i)} \cdot cooc(e, f; E, F) \tag{2.8}$$

where $cooc(e, f; E, F)$ is the number of times e and f co-occur in the sentence pair (E, F). Mathematically, the co-occurrence count for two words in a sentence pair is:

$$cooc(e, f; E, F) = \left(\sum_{i=0}^{n} \delta(e, e_i)\right)\left(\sum_{j=1}^{m} \delta(f, f_j)\right) \tag{2.9}$$

that is, if a word type e occurs x times in E, and f occurs y times in F, we say that e and f co-occur xy times in (E, F). In our sample alignment in Figure 2.1, the types *the* and

le co-occur four times. Equations 2.5 and 2.8 have been introduced in terms of counts of links or co-occurrences. These counts have in turn been described in terms of Kronecker delta function sums over sentences. The delta function notation is needed for more complicated Candide models, while the counting notation is useful because it demonstrates that, at a fundamental level, the Candide models are primarily powered by counting word co-occurrences. This reliance on co-occurrence and link counts will tie together all reviewed models for word alignment, including the novel model introduced in this thesis.

Equation 2.8 will yield the exact same counts as Equation 2.5, but the former needs to consider only the $n+1$ possible generators of $\mathbf{f}$, and not the $(n+1)^m$ possible alignments for the sentence pair. The time complexity of this method for collecting a single count is $O(m+n)$, while the previous counting method required $O((n+1)^m)$ time. This fast collection of counts is one of Model 1's largest advantages, as it allows correct estimates over all possible alignments inexpensively. This will not be possible with some of the more complicated models.

In summary, the EM algorithm begins by guessing parameters that will induce a probability distribution over alignments given a sentence pair. The system then counts fractional alignment links based on the current distribution. It uses these counts to calculate new parameters for the distribution, and the process begins again. In our description of EM, all that has been guaranteed is that the likelihood of the data will continue to improve until a local maximum is reached. A system like EM can be sensitive to the initial settings of the distribution parameters (the $P_t(f|e)$ table). Fortunately, the parameter space induced by Model 1 has a unique local maximum. This makes the selection of starting parameters irrelevant. Though the algorithm described here begins with flat parameters, any initial values where $\forall(e,f)|P_t(f|e) \neq 0$ would also be valid.

With the training and modeling processes described, all that remains is our primary goal: the task of finding the most probable alignment given the model. With Model 1's assumptions in place, this process is very simple, so simple that it should highlight some of Model 1's weaknesses. The algorithm for finding the most probable alignment is as follows:

- Given a sentence pair (E,F) where F is of length m:
- For $1 \leq j \leq m$
  - Select an English word $e_i \in E$ such that $P_t(f_j|e_i)$ is maximized. Break ties arbitrarily.
  - Set $a_j = i$ in A.

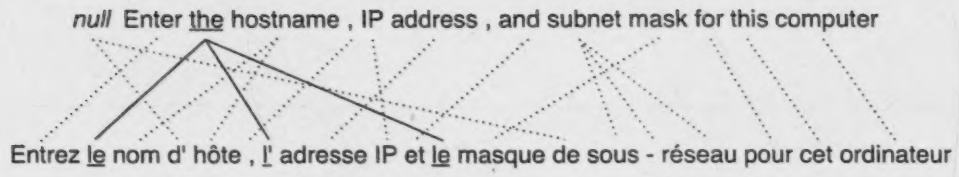

null Enter the hostname , IP address , and subnet mask for this computer
Entrez le nom d' hôte , l' adresse IP et le masque de sous - réseau pour cet ordinateur

Figure 2.2: An Example Candide Model 1 Alignment

As was the case with counting, the simplicity of this algorithm stems from the fact that the selection of a link in A cannot affect the probability of any other link in A. One can imagine that such an algorithm does not always generate satisfying results. Depending on the tie-breaking mechanism, it is quite likely that for word types that occur as multiple tokens in E, a single token of that type can generate every instance of the French translation in F. A similar problem is illustrated in Figure 2.2, which shows a Candide Model 1 alignment of a relatively simple sentence pair. The problem occurs with the word, *the*. Both *le* and *l'* are correct dictionary translations of *the*. It is unlikely, though, that all three *the* translations correspond to the single English *the* token. Also note how the model has no notion of locality or position. A link between *the* and *le* where both are near the beginning of their respective sentences is as good as a link between *the* and *le* where one is near the beginning and the other is in the middle. As the models become more complicated, the alignments will become less susceptible to these sorts of problems. The next problem to be addressed will be issue of position in alignment.

2.2.3 Candide Model 2

Candide Model 2 looks very much like Candide Model 1. It alters only the position selector, making it take French and English positions into account as well as sentence lengths. This maintains some of Model 1's more attractive properties, while extending the model to use more information. To create the new model, the following terms in Equation 2.3 are altered:

1. As in Model 1, $P(m|E)$ is simplified as ϵ.

2. $P(a_j|a_1^{j-1}, f_1^{j-1}, m, E)$, the position selector, is assumed to depend on the sentence lengths and the English and French positions participating in a_j. It becomes the distribution $P_a(a_j|j, m, n)$.

3. As in Model 1, $P(f_j|a_1^j, f_1^{j-1}, m, E)$, the French type selector, is simplified to $P_t(f_j|e_{a_j})$.

The final equation for Candide Model 2 probability is:

$$P(F, A|E) = \epsilon \cdot \prod_{j=1}^{m} P_t(f_j|e_{a_j}) P_a(a_j|j, m, n) \tag{2.10}$$

Model 2 introduces a new table to the modeling system. This alignment table stores the likelihood of an English position generating a word, given the position of the French word and French and English sentence lengths. For example, $P(1|2, 5, 6)$ is the probability that the first English word generated the second French word in a sentence pair where the French sentence is 5 words long and the English sentence is 6 words long. As $P_a(a_j|j, m, n)$ is a probability distribution, it must be the case that for any fixed (j, m, n), the probabilities for all possible values of a_j must sum to 1.

Model 2's two probability tables, P_t and P_a, will be trained using the EM algorithm. Unfortunately, the parameter space specified by Model 2 no longer has a unique local maximum. Therefore, Candide bootstraps Model 2 by initially using Model 1 probabilities instead of a flat distribution. It is hoped that this will begin the search through parameter space closer to the objective, the global maximum. In order to bootstrap, the first iteration of Model 2 EM calculates counts using Model 1's table for $P_t(f|e)$, and using $P_a(i|j, m, n) = \frac{1}{n+1}$.

The counts needed for Model 2 parameters could be acquired by enumerating all possible alignments for a sentence pair, and weighting those alignments by their probability under the current model. Our naïve counting method for $c(f|e; F, E)$ remains the same as in Equation 2.5, while the counts for the P_a table are acquired similarly:

$$c(i|j, m, n; F, E) = \sum_{A \in \mathcal{A}} P(A|E, F) \cdot \delta(i, a_j) \tag{2.11}$$

The delta function used in Equation 2.11 returns one only when the French word at position i is linked to the English word at position j in A.

Once again, there are far too many alignments to enumerate each explicitly. However, there is still no term in the Model 2 $P(F, A|E)$ that takes into account the links that have been previously established in A. As with Model 1, this allows each potential link in an alignment to be counted independently. As two factors now affect the likelihood of a link, the improved counting equations are a little more complicated and a little more costly than in Model 1, but they remain vastly superior to enumerating every possible alignment. The equations have little bearing on the results presented in this thesis, so they have been omitted. It is enough to know that an efficient way to implicitly count all possible alignments

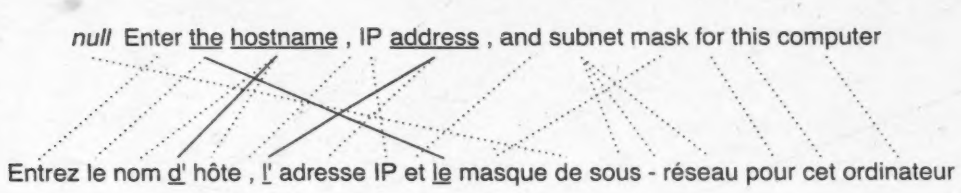

null Enter the hostname , IP address , and subnet mask for this computer
Entrez le nom d' hôte , l' adresse IP et le masque de sous - réseau pour cet ordinateur

Figure 2.3: An Example Candide Model 2 Alignment

exists. Once parameters have been set by EM, the most likely alignment can be found with an algorithm similar to Model 1's alignment generation algorithm:

- Given a sentence pair (E, F) where F is of length m:
- For $1 \leq j \leq m$
  - Select an English word e_i from E such that $P_t(f_j|e_i) \cdot P_a(i|j, m, n)$ is maximized.
    Break ties arbitrarily.
  - Set $a_j = i$ in A.

As before, each French word's generator is selected independently, so no information regarding other establish links can be taken into account. However, the positional awareness provided by $P_a(i|j, m, n)$ does produce more sensible alignments.

Figure 2.3 shows the most likely alignment for our example sentence from Figure 2.2 under Model 2 scoring. One can see several improvements: the *d'* in *nom d' hôte* is now aligned correctly to *hostname*, where before it was attributed to the *null* word. The French word *l'* also exhibits an improvement. It makes more sense to attribute the spurious determiner *l'* to the English *address*, as *address* produces the French *adresse*, which in turn requires *l'*, than it does to attribute it to the English *the*, which is simply incorrect in this context. Unfortunately, the incorrect link between *the* and the second *le* remains in place.

Models 1 and 2 introduce two terms in the estimation of $P(F, A|E)$ that are both well-behaved and useful: type to type information is stored in the translation table P_t, and position to position information is stored in the alignment table P_a. These tables, or tables like them, will be used throughout the remaining Candide models. The problem of not considering the other links in the alignment when determining the likelihood of a new link still remains. This is the next major area targeted for improvement. The two following models explore two different ways existing links can be taken into account.

2.2.4 HMM Alignment

First introduced in [41] and refined in [36], the Hidden Markov Model (HMM) for word alignment borrows ideas from speech recognition in order to capture the notion of locality between the links in an alignment. An HMM is a first-order statistical model that models a sequence of observations as if they were produced by a path through some unobservable finite state machine with emitting states [39]. Much of the theory associated with HMMs is unnecessary for the word alignment task; for example, an actual finite state machine is very rarely drawn and explained when discussing such models. Instead, the dynamic programming algorithms associated with HMMs are used to work with a first order simplification of Equation 2.3 in which the following terms are altered:

1. As in Model 1, $P(m|E)$ is simplified as ϵ.
2. $P(a_j|a_1^{j-1}, f_1^{j-1}, m, E)$, the position selector, is assumed to depend on the position of the link connected to the previous French position according to A and the English sentence length n. It becomes the first order distribution $P_a(a_j|a_{j-1}, n)$.
3. As in Model 1, $P(f_j|a_1^j, f_1^{j-1}, m, E)$, the French type selector, is simplified to $P_t(f_j|e_{a_j})$.

The final equation for HMM probability is:

$$P(F, A|E) = \epsilon \cdot \prod_{j=1}^{m} P_t(f_j|e_{a_j}) P_a(a_j|a_{j-1}, n) \tag{2.12}$$

One can see that the HMM over-rides the position selector introduced in Model 2. In fact, the HMM has completely replaced Model 2 in the state of the art Candide implementation, GIZA++ [37]. This new formulation for the position selector is designed to capture information regarding locality in alignments. Generally, words close to each other in the source language tend to remain close to each other in the translation. This is most easily illustrated when the alignment is drawn using a grid format, as in Figure 2.4. In this figure, a grey dot in a grid position indicates that the French word on the x-axis is aligned with the corresponding English word on the y-axis. Due to the Candide generative story, only one dot can exist in each column, as each French word is generated by exactly one English word. Alignments to *null* have been omitted for the purposes of this illustration. The alignment pictured is the correct alignment for our example sentence pair.

In Figure 2.4, it appears as if links tend to follow certain geometric patterns. Most often, as you move from left to right along the grid, the links move in diagonal steps, or in straight lines. The purpose of the $P_a(a_j|a_{j-1}, n)$ table is to learn this geometric bias to the extent

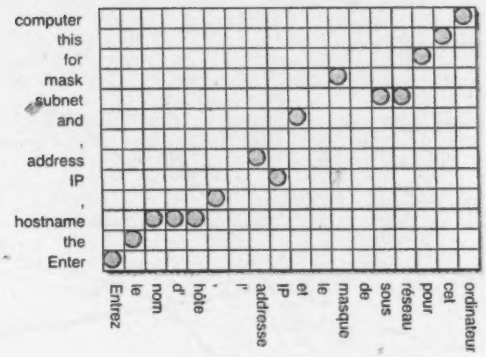


Figure 2.4: An HMM Alignment in Grid Format

that it exists in the training data. To keep computation feasible, only first-order patterns are used, that is, each link is influenced only by the link in the column directly before it.

We can view Figure 2.4 as a graphical representation of a path through the conceptual HMM that generated the French sentence. The French words are treated as observables, where positions in the French sentence are time-steps in our series. Each row in the grid is a state, and each state can transition to any other state in the model. The event at position (1,1) in the grid occurs when the state $Enter_1$, representing the English sentence position 1, generates the French word *Entrez* in time step 1. In other words, the grid position shows that *Entrez* was linked to *Enter*. Due to the nature of the Candide models, the HMM transition and emission tables are not indexed by state, as in a normal HMM. Instead, the probabilities are read directly from the P_a and P_t look-up tables according to state characteristics (ie: the state's position indicator, or its English word label). The alignment table, P_a becomes like a transition table, it answers the question, "If the machine was in $Enter_1$, what is the chance of transitioning to the state the_2?" Notice how the transition probability depends only on the previous state, and not on the observable emitted in that state. The translation table, P_t is like an HMM emission table. It answers the question, "If the machine is currently in the state the_2 what is the chance of emitting the observable *le*?"

With HMM alignment, we have our first model that takes existing links into account when making a new link. Remarkably, even though links are no longer completely independent of other links, we can still implicitly enumerate all possible alignments efficiently, allowing the collection of correct parameter counts. A dynamic programming procedure known as the forward-backward algorithm [39] exists to efficiently find the probability of being in a particular state (English sentence position) at a particular time step (French sen-

tence position) when given model parameters (P_a and P_t). Armed with this technique, one can calculate the probability of each possible link individually, without enumerating the exponential number of possible alignments. These probabilities can be used to get the fractional counts needed for the look-up tables. Details of the forward-backward algorithm are beyond the scope of this thesis, but more details can be found in [39]. The HMM training process is still EM, it only differs from Candide models 1 and 2 in its P_a position table and its clever link counting mechanism. There is no unique local maximum in the HMM parameter space. Therefore, one should bootstrap the HMM training process using model 1 alignments in order to give the algorithm a good starting point.

The most likely alignment given a sentence pair and model parameters can be found using another algorithm lifted from HMM literature. The Viterbi algorithm [39] is a dynamic programming procedure used to find the most likely path through an HMM given an observation sequence. For historical reasons, the most likely word alignment given a model is often referred to as the Viterbi alignment, regardless of whether or not the underlying model is an HMM. The Viterbi algorithm uses a recursive definition of path probability in order to calculate the likelihood Q of the most likely partial path through the model that passes through state i at time step j:

$$Q(i,j) = P_t(f_j|e_i) \cdot \max_{0 \leq i' \leq n} [P_a(i|i',n) \cdot Q(i',j-1)]$$

A dynamic program that uses this recurrence relation along with backtracking pointers can efficiently build the most likely path through the model. It will implicitly enumerate all possible alignments. This algorithm, though more costly than the alignment algorithms for Candide models 1 and 2, is instinctively more appealing, as we take past decisions into account when making a new link.

There are still a number of technical issues that must be dealt with before a working HMM alignment system can be integrated into the Candide models. First of all, it is not clear that getting alignment probabilities based entirely on the positions a_j and a_{j-1} is advantageous. Our motivating example showed that it was the current position relative to the previous position that was consistent in our alignment. Considering this motivation, there is very little difference between a link to 4 whose previous link was to 3, and a link to 14 whose previous link was to 13. One could generalize, and call both links $14 - 13 = 4 - 3 = +1$. This is done in both [41] and [36]. This creates an approximation to $P_a(a_j|a_{j-1},n)$, represented by the unnecessarily small table $P_a(a_j - a_{j-1}|n)$. To take advantage of the space made available by this approximation, [36] conditions on the word classes involved as well

as English sentence length, creating a table $P_a(a_j - a_{j-1}|C(e_{a_j}), C(f_j), n)$. The inclusion of these two new class-based conditions is valid. This new table actually uses more conditioning factors from the original alignment Equation 2.3 for $P(F, A|E)$ than Equation 2.12. In [36], they use 50 classes for both French and English words. The classes are generated from the training data using an external clustering program. As a final modification, the *null* word was originally given the arbitrary English position 0. Under this new difference-based model of P_a, 0 takes on an unintended meaning. To counteract this, $n - 1$ extra *null* states are added in [36] so that a link to a *null* word also encodes the position of the word linked before the *null* word occurred. The details of these technical issues are not important. They are mentioned to demonstrate the tweaks and innovations required to make an originally elegant probabilistic model feasible for use in practice. Having examined this first-order model for considering past links, we now turn back to the original Candide system to examine a less elegant, but much more ambitious attempt to take past links into account.

2.2.5 Candide Model 3

The HMM model allows us to consider other links in an alignment. However, basing our decisions entirely on the position of the word linked to the previous French position still seems fairly myopic. What other aspects of existing links would be useful to consider, without having computation explode into intractability? The Candide models allow for one English word to generate several words in the French sentence. It stands to reason that these words will be related to the word that generated them and to each other. A translation model could benefit from awareness of what other words a candidate English word has generated when considering a new link. For example, the model could be made aware that a single instance of the English word *the* rarely generates more than one word in the corresponding French sentence. This would make the alignment shown and criticized earlier in Figure 2.2, with three words linked to a single *the* token, less likely.

It would be useful to model the interactions between the French words generated by a single English position directly, to capture some of our natural intuitions of how translation works. Unfortunately, with the base decomposition defined in Equation 2.3, it is difficult to see where a term for dealing with the relationship between words generated by a given English word would fall out. This is because Equation 2.3 is focused on the French sentence. It steps through the French sentence and selects a generator for each French word. The notions described here would be easier to capture with a new decomposition that is aware of all the words an English word has generated. This decomposition would instead

be focused on the English sentence. Candide Model 3 and Model 4 introduce a new base decomposition with qualities that make dealing with multiple French words generated by the same English word easier. As was done with Equation 2.3, this base decomposition will be then be simplified to different degrees for Models 3 and 4.

A new decomposition entails a new generative process that describes how F is created from E. The process that is used as a basis for Candide Models 3 and 4 begins by selecting a fertility for each English word, including *null*. The fertility ϕ_i of e_i is the number of words that e_i will generate in the French sentence. $\sum_{i=0}^{n} \phi_i = m$ becomes the length of the French sentence. For each English word e_i, the French words to be generated are selected in the form of a tableau τ_i, which is an ordered sequence containing ϕ_i French words. The words in the tableau are then placed in the French sentence according to a permutation π_i, which is a sequence of ϕ_i French sentence positions. The *null* word always waits until all other words have selected their values for ϕ and π before selecting its own. We will use $\tau_{i,k}$ to represent the k^{th} element in the i^{th} word's tableau. Similarly, $\pi_{i,k}$ represents the k^{th} position in the i^{th} word's permutation.

This generative process can be used to create an equation for the probability of a tableau-permutation pair (τ, π):

$$
\begin{aligned}
P(\tau, \pi|E) = & \prod_{i=1}^{n} \left[P(\phi_i|\phi_1^{i-1}, E)\right] \cdot P(\phi_0|\phi_1^n, E) \\
& \cdot \prod_{i=0}^{n} \left[\prod_{k=1}^{\phi_i} \left[P(\tau_{i,k}|\tau_{i,1}^{k-1}, \tau_0^{i-1}, \phi_0^n, E)\right]\right] \\
& \cdot \prod_{i=1}^{n} \left[\prod_{k=1}^{\phi_i} \left[P(\pi_{i,k}|\pi_{i,1}^{k-1}, \pi_1^{i-1}, \tau_0^n, \phi_0^n, E)\right]\right] \\
& \cdot \prod_{k=1}^{\phi_0} \left[P(\pi_{0,k}|\pi_{0,1}^{k-1}, \pi_1^n, \tau_0^n, \phi_0^n, E)\right] \qquad (2.13)
\end{aligned}
$$

This equation represents the probability of one of several tableau-permutation combinations that could create a given sentence F according to an alignment A. As with our previous base equation (Equation 2.3), this decomposition is completely correct, and has yet to make any simplifying assumptions. To use this decomposition as a translation model to find alignments, the probability we are interested in is $P(F, A|E)$. This can be found by summing every (τ, π) pair that would result in an alignment A. Let $\langle F, A\rangle$ be the set of all (τ, π) terms that would generate a given (F, A) pair. Then we define probability of (F, A)

as:

$$P(F, A|E) = \sum_{(\tau,\pi)\in\langle F,A\rangle} P(\tau, \pi|E) \tag{2.14}$$

There are $\prod_{i=0}^{n} [\phi_i!]$ elements in $\langle F, A\rangle$, as each individual (τ_i, π_i) pair can be arranged $\phi_i!$ ways to produce the same effect.

Once $P(F, A|E)$ is established, Equation 2.2 for finding $P(A|E, F)$ still stands, as it is decomposition-independent. This base decomposition is larger and less intuitive than the base decomposition described in Equation 2.3. Fortunately, it will be simplified to more familiar terms before being used in practice. The important thing to notice about this decomposition is that it has teased out terms that encapsulate the factors we are interested in. The first line in Equation 2.13 deals with fertilities, that is, an English word's probability of generating a specific number of French words. The second line connects French type selection to the other French types that have been generated by the current English word. The third and fourth lines connect the selection of French positions to the previous positions generated by this English word. We are now free to ignore any of the conditionalizing terms for the purposes of simplification.

Model 3 will essentially operate in the same way as Model 2, but with an extra term modeling English word fertility. To create this model, the following terms in Equation 2.13 are simplified:

1. For $1 \leq i \leq n$, the fertility probability $P(\phi_i|\phi_1^{i-1}, E)$ depends only on the fertility in question, ϕ_i and the type of the word in question, e_i. It becomes $P_f(\phi_i|e_i)$.

2. The tableau term $P(\tau_{i,k} = f_j|\tau_{i,1}^{k-1}, \tau_0^{i-1}, \phi_0^n, E)$ depends only on $\tau_{i,k}$ and the generating English word e_i. It becomes the same as Model 2's translation table $P_t(f_j|e_i)$.

3. The permutation term $P(\pi_{i,k} = j|\pi_{i,1}^{k-1}, \pi_1^{i-1}, \tau_0^n, \phi_0^n, E)$ depends only on the position of the French token, $\pi_{i,k}$, the position of the English generator, i, and the lengths of the two sentences, n and m. It becomes very similar to the position selector from Model 2, but with the English position (i) and French position (j) terms inverted: $P_a(j|i, m, n)$.

4. The *null* word is handled completely differently. *null* is assumed to place words only after all other words are placed, and thus, only the gaps remaining in the sentence have a non-zero probability for word placement. If we assume that these positions are all equally likely, regardless of the word being placed, this leads to the $\prod_{k=1}^{\phi_0}(\pi_{0,k}|\pi_{0,1}^{k-1}, \pi_1^n, \tau_0^n, \phi_0^n, E)$ term contributing $\frac{1}{\phi_0!}$ for all the words in τ_0. We

will continue to use translation probability of $P_t(f|e = null)$ to get *null* type probabilities. This leaves only a term for ϕ_0, the fertility of *null*. To take into account the notion that longer sentences should have a greater number of *null* generated words, *null* words are modeled as having a certain probability p_1 of occurring after any given French word. With this assumption in place, and a second term p_0 defined as the probability of the *null* word not occurring ($p_0 + p_1 = 1$), we can simplify $P(\phi_0|\phi_1^n, E)$ as follows:

$$P(\phi_0|\phi_1^n, E) = \binom{\phi_1 + \cdots + \phi_n}{\phi_0} p_0^{(\phi_1+\phi_2+\cdots+\phi_n-\phi_0)} p_1^{\phi_0}$$

Taking Equation 2.14 into account, this creates the following Model 3 central equation:

$$P(F, A|E) = \binom{m - \phi_0}{\phi_0} p_0^{(m-2\phi_0)} p_1^{\phi_0} \cdot \prod_{i=1}^{n} [\phi_i! \cdot P_f(\phi_i|e_i)] \cdot \prod_{j=1}^{m} \left[P_t(f_j|e_{a_j})\right] \cdot \prod_{j=1:a_j\neq 0}^{m} [P_a(j|a_j, m, n)] \quad (2.15)$$

One can see that Model 3 is no longer very elegant. It does, however, account for English word fertilities, and it incorporates a special p_1 parameter for dealing with the fertility of *null*. Otherwise, it is not so different from Model 2. The only other major difference is the inversion of i and j in the position selector, which falls out of the new generative story. The parameters needed for Model 3 are the tables P_t, P_a and P_f, as well as the single *null* fertility parameter p_1.

Model 3 is, however, a very different animal from the models reviewed earlier in this thesis, as the fertility parameter ties all of the links in a given alignment together. We can no longer determine the likelihood of a generator for any given French word independently. There is no known method to implicitly (and efficiently) enumerate all alignments under Model 3. For the purposes of training and finding the most likely alignment, we are left with the daunting task of enumerating all possible alignments explicitly. Since this is not tractable with current computing technology, we will instead estimate the results of an enumeration heuristically. First, we will describe a method to estimate the most likely alignment given a set of Model 3 parameters. This method can then be used to help sample from the space of possible alignments, in order to simulate enumerating the entire distribution $\mathcal{A}$. This sampling procedure will allow us to do Model 3 training. The initial Model 3 parameters needed by both processes are provided by special-purpose algorithms designed to transfer Candide Model 2 or HMM distributions into Model 3 parameter space [4, 36]. The details of these transfer algorithms are beyond the scope of this thesis.

Our first task is to estimate the most likely alignment for a sentence pair, given Model 3 parameters. To do so, we will begin with the most likely alignment according to a Model 2 or HMM distribution, which can be found easily. A search algorithm will then take a number of greedy steps away from the starting alignment, such that each step provides a maximal increase to the Model 3 formulation for $P(F, A|E)$. The search continues until an alignment is reached from which no single step can increase Model 3 probability. A step is defined as either a move, where a single French word selects a new English generator, or a swap, where two French words swap their respective English generators. The probability score for a new alignment generated by a single step can be calculated efficiently using the previous alignment's probability and a constant-time incremental update equation. The greedy alignment produced in this matter is taken as an estimate of the most likely, or Viterbi alignment for this sentence pair.

Given a greedy Viterbi alignment for a sentence pair, we can sample from the alignment distribution $\mathcal{A}$ in order to simulate enumerating the entire distribution. To do so, we will take advantage of the fact that, in later models, the majority of the probability mass is consumed by a relatively small number of highly probable alignments [4]. Assuming that this Viterbi alignment is one such highly probable alignment, we can simply sample from the alignments that are close to it in alignment space. Hopefully this sample will represent a significant fraction of the true probability mass. There are two main approaches to sampling the space of probable alignments, both are fairly effective. The first involves a procedure known as pegging, where one French word in the base alignment is frozen with its current generator. The greedy alignment then proceeds as usual, leaving that one French word untouched. In effect, it estimates the most likely alignment where that one link remains unchanged. This sampling method, originally described in [4] takes a sample of $m + 1$ alignments. The sample consists of the Viterbi alignment described in the paragraph above, as well as one greedy alignment for each of the m French words, where that word's link in the base alignment was pegged in place. The second method, suggested in [1] and implemented in [37], samples from the space directly around the Viterbi alignment. Its sample consists of every alignment that can be arrived at in one step (defined above) from the Viterbi alignment. This method is very fast, as each alignment's probability can be calculated in terms of the base alignment probability with a constant-time incremental equation. It has been shown in [37] that pegging is only slightly more effective than step-based sampling. Both methods have been shown to be significantly more effective than taking a sample consisting of only the Viterbi alignment.

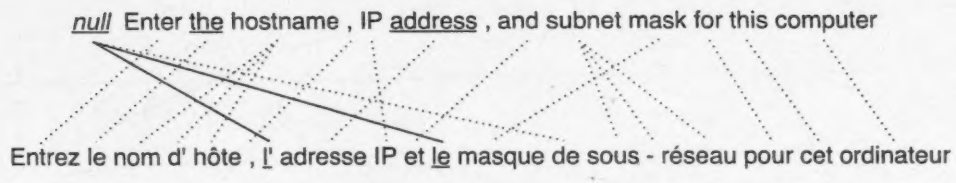

null Enter the hostname , IP address , and subnet mask for this computer
Entrez le nom d' hôte , l' adresse IP et le masque de sous - réseau pour cet ordinateur

Figure 2.5: An Example Candide Model 3 Alignment

Regardless of the sampling method used, given a sample, one can get estimates of counts for Model 3 parameters. These counts are taken by counting links from the alignments in the sample, where each link has a fractional value equal to the probability of the alignment. Let $\mathcal{A}_s$ be our sample of $\mathcal{A}$. Each link in an alignment A from $\mathcal{A}_s$ is given the value of:

$$P(A|E,F) \approx \frac{P(F,A|E)}{\sum_{A' \in \mathcal{A}_s} P(F,A'|E)} \tag{2.16}$$

That is, each link in an alignment A gets a count equal to $P(F,A|E)$, normalized so that $\sum_{A \in \mathcal{A}_s} P(A|E,F) = 1$. The equations which formally define these counts, and their corresponding probability table values are very similar to those used to describe the naïve counts and tables from Models 1 and 2, such as Equations 2.5 and 2.7. These equations are not included here, as their forms are quite intuitive.

We are now fully equipped to train Model 3, and to use it to get improved word alignments. Model 3 represents a significant improvement over Model 2, due to its awareness of links which share the same English word. Figure 2.5 shows our example sentence pair aligned according to Model 3. This is also the correct alignment for this sentence pair, by most standards. The link for *l'* is now pointed at *null*; in Model 2, it was linked to *address*. Both may be considered correct, though the Model 3 link is preferred by our alignment standards. In Figure 2.5, *the* is finally generating only one word, as the second *le* is now being generated by *null*. Otherwise, the alignment remains unchanged from Model 2. This example represents a fairly easy sentence pair; many sentence pairs do not achieve complete accuracy, even after Model 4.

It was shown in [37] that Model 3 is not much more accurate than an HMM. In fact, the state-of-the-art Candide implementation, GIZA++ generally skips Model 3 altogether when moving through the Candide model hierarchy. Fortunately, the theoretical work done in establishing Model 3 will make for an easy leap to Model 4.

2.2.6 Candide Models 4 and 5

The final Candide model to be discussed in detail here is Candide Model 4. This model considers all of the French words generated by a single English word to be a phrase. It explicitly models how the words in a phrase are placed with respect to the previous French phrase, and with respect to other words in the current phrase. In this way, it is hoped that the model will allow large French phrases to move around during translation with respect to their English generators without excessive penalty. As in Model 3, the *null* word will be considered an exception. The words it generates will not be considered phrases.

To accomplish its modeling goals, Model 4 makes a number of simplifications to Equation 2.13. Before we can discuss those modifications, we will have to define some terminology. From Model 3, we know that the ordered sequence of French words generated by an English word e_i is called its tableau τ_i, and these words are placed according to a permutation π_i. We define the center, $\odot_i$, as the average of the positions of the French words generated by e_i, rounded up. Formally, we say:

$$\odot_i = \left\lceil \frac{\sum_{k=1}^{\phi_i} \pi_{i,k}}{\phi_i} \right\rceil$$

Additionally, we define the string head of e_i to be the French word in τ_i that has the French sentence position with the lowest numerical value. The string head of e_i is effectively the left-most word in F placed by e_i. We will assume that τ_i and π_i are ordered so that $\pi_{i,k} > \pi_{i,k-1}$. That is, the sequences will list words and positions in their left-to-right order in the French sentence. Because of this, we will consider only one of the $\phi_i!$ possible orderings of τ_i. With this ordering in place, e_i's string head is always the French type $\tau_{i,1}$ and is placed in position $\pi_{i,1}$. After defining these terms, Model 4 can make the following simplifications to Equation 2.13:

1. The fertility term is simplified using P_f as in Model 3.

2. The tableau term is simplified using P_t as in Model 3.

3. The permutation term $P(\pi_{i,k} = j|\pi_{i,1}^{k-1}, \pi_1^{i-1}, \tau_0^n, \phi_0^n, E)$ is calculated using two equations, one for the string head and one for the other words in τ_i. For the string head, probability is taken to depend on the difference between the proposed position and the center of the previous English word, as well as the classes of the previous English word and the proposed French word. The classes are assigned according to the same 50 clusters used in HMM probability calculation. This creates a table of the

form:

$$P_{a,1}(j - \odot_{i-1} | C(e_{i-1}), C(f_j))$$

For other words in the tableau, probability depends on the difference between the proposed position and the position of the previous word in the tableau, as well as the class of the proposed word. This creates a table of the form:

$$P_{a,>1}(j - \pi_{i,k-1} | C(f_j))$$

4. The *null* word is dealt with using p_1 as in Model 3.

This creates the following Model 4 central equation:

$$P(F, A|E) = \binom{m - \phi_0}{\phi_0} p_0^{(m-2\phi_0)} p_1^{\phi_0} \cdot \prod_{i=1}^{n} [P_f(\phi_i|e_i)] \cdot \prod_{j=1}^{m} [P_t(f_j|e_i)] \cdot \prod_{i=1:\phi_i \neq 0}^{n} \left[\begin{array}{l} P_{a,1}(\pi_{i,1} - \odot_{i-1} | C(e_{i-1}), C(f_{\pi_{i,1}})) \\ \cdot \prod_{k=2}^{\phi_i} \left[P_{a,>1}(\pi_{i,k} - \pi_{i,k-1} | C(f_{\pi_{i,k}})) \right] \end{array} \right] \quad (2.17)$$

Assuming that the clustering algorithm used to create the C functions for French and English words is capable of picking out clusters with syntactic meaning, Model 4 is capable of considering some syntactic regularities when it comes to word placement. For example, the $P_{a,1}$ term can conceivably learn that most French adjectives have their positions with respect to their noun reversed during translation. Take, for example, the fact that the phrase "quick$_1$ fox$_2$," translates to, "renard$_1$ (fox) rapide$_2$ (quick)" and not, "rapide$_1$ renard$_2$". This would be stored by making the entry that covers *renard* being placed ($f_j = renard$) by an English word that follows *quick* ($e_{i-1} = quick$), where the position of the new English word is one position before center of *quick*'s translation ($j = 1$ and $\odot_{i-1} = 2$, therefore $j - \odot_{i-1} = -1$), $P_{a,1}(-1|C(quick), C(renard))$ greater than the entry where *renard* is placed in the position after *quick*'s translation, $P_{a,1}(+1|C(quick), C(renard))$. This is not so different from the abilities of the HMM, but extra information is added due to phrasal awareness. The $P_{a,>1}$ term allows the model to learn how phrases fit together, for example, it can learn whether or not all the words in a phrase should be consecutive, or if there should be interruptions by words from other phrases.

Model 4 training and alignment algorithms are nearly identical to those used in Model 3. Once again, there is no way to efficiently enumerate all possible alignments, making heuristic search necessary for finding the most likely alignment, and sampling necessary in training. Since Model 4 is significantly more complicated than Model 3, and requires active awareness of all words generated by an English word, the incremental probability

update equations used for alignment search and training are not nearly as fast as in Model 3. Beyond slight changes made to increase efficiency, the processes are the same as in Model 3.

The only Candide Model that remains to be described is Model 5. Model 5 introduces no new linguistic intuitions, nor does it take advantage of additional information. Instead, it improves upon Models 3 and 4 by eliminating deficiency. Models 3 and 4 are considered deficient because they assign non-zero $P(F, A|E)$ probabilities to sentences that are not possible. This problem arises with the simplifications made to Equation 2.13; it is not present in the base model itself. The impossible sentences occur because Models 3 and 4 have no safeguards to prevent an English sentence from placing multiple words in its French translation on the same French position. Under the generative process described by these models, the English sentence, "The dog jumps" could generate the correct French words, *Le*, *chien*, and *saute* and then place them all on position 1 of the French sentence, leaving positions 2 and 3 vacant. A deficient model can improve its probability during EM iterations without improving visible performance by assigning less mass to the impossible events and redistributing the liberated mass evenly among possible events.

IBM Model 5 is a version of Model 4 designed to eliminate deficiency by using override probabilities to assign an event probability zero if it has two words sharing the same position. This alteration makes calculating $P(F, A|E)$ even more complicated, and greatly slows the learning process. It was shown in [37] that one major problem with Candide Models 3 and 4 with respect to deficiency is that the terms that make up the model are not equally deficient. The term for real words is deficient, while the term used for the *null* word is not. This produces strange effects as EM places more emphasis on the *null* word in order to increase overall probability. They present a simple fix to make Model 3 and 4's *null* term deficient as well. It was shown in [37] that with this fix in place to weaken Model 4's *null* term, transitioning from Model 4 to 5 actually decreased accuracy. Model 5 will not be discussed further in this thesis.

2.2.7 GIZA++ and Model-Independent Extensions

There have been several implementations of the Candide model for machine translation. The first was a closed project by IBM, documented in [4]. The second implementation took the form of a group effort by the academic machine translation community. Models 1 through 4 were implemented with little deviation from what is described in [4] as a component known as GIZA of a complete machine translation package dubbed EGYPT [1]. The

source code was made available to the wider community in order to encourage academic exploration of the ideas introduced by IBM. The latest implementation is dubbed GIZA++ and is documented in [36] and [37]. GIZA++ added an HMM model to the package, and improved Models 3 and 4 as described above. It also implements a smoothing method to aid with parameter over-fitting and rare words. Some of these methods are specific to using German as a target language (corresponding to English in all of the explanations in our Candide descriptions). They elected to smooth only the position and fertility terms in $P(F, A|E)$. Position was smoothed by using linear interpolation with a constant distribution. In a German-specific modification, fertility was smoothed by interpolating with a second distribution conditioned on the length of a word as opposed to a word's type.

The work accompanying GIZA++ also developed a scoring metric for alignment quality based on comparison to a hand-made reference alignment. This scoring metric has since become a community standard, and will be described when it is used to test the novel alignment method developed in this thesis. This scoring metric allowed the GIZA++ team to determine which models were contributing to the alignment process, and which were detracting from it. They found the most advantageous model sequence for their corpora and reference alignments to be Model 1, transitioning to an HMM, and finishing with Model 4, skipping both Models 2 and 3 [37].

Finally, [37] also introduced model-independent extensions that can be used to enhance an alignment produced by any Candide model. These extensions were motivated by the fact that Candide can make many-to-one links in only one direction, from English to French. An English word can link to several French words, while a French word can only link to one English word. The translators creating reference alignments for evaluation often felt it necessary to make many-to-one links in both directions. Sometimes a single French word did occur because of more than one of the English words in the corresponding sentence. In response to this, a method to produce alignments with many-to-one links in both directions using Candide was developed. This extension made no changes to the models themselves. Instead it involved combining two sets of alignments, one created from a model trained with French as the source and English as the target, and a second model created by using English as the source and French as the target. Each of these alignments would be capable of creating many-to-one links in a specific direction. Three methods were developed to combine a pair of alignments: intersection, union, and refined combination.

In intersection, the only links present in the final alignment are the links present in both of the input alignments. This produces very precise alignments with low recall: most of the

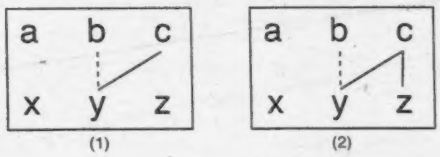


Figure 2.6: An Example of Refined Alignment Combination

links made by the system are correct but fewer correct links are found. Also, intersection cannot produce anything but one-to-one alignments, many-to-one links are eliminated in both directions. Union takes links that occur as a result of either of the input alignments. This finds most of the correct links, but sacrifices precision: many of the links in the new alignment are not correct. Refined combination is designed to reach a compromise between precision and recall. It begins with intersection, and then adds links from the union only if they meet at least one of two criteria. If a prospective link connects two words that are both unlinked in the new alignment, then it is added. Failing that, a link l can also be added if another link l' exists such that it would be either a source or target neighbor to l. A link $l(i,j)$ connecting the word in position i in the target language to j in the source is considered to have a source neighbor if another link $l'(i+1,j)$ or $l'(i-1,j)$ exists. Similarly, $l(i,j)$ has a target neighbor if $l'(i,j+1)$ or $l'(i,j-1)$ exists. In this case, a link from the union can be added so long as it would have a target or source neighbor, and so long as it does not cause any link to have both target and source neighbors. For example, in Figure 2.6, the dotted link $l(b,y)$ in the alignment labeled (1) would be allowed, as it has a source neighbor $l(c,y)$. It would not be allowed in (2), as its addition would cause the link $l(c,y)$ to have both a source neighbor $l(b,y)$ and a target neighbor $l(c,z)$. It has been shown in [37] that refined link combination does strike an acceptable balance between link precision and recall.

GIZA++ extended with refined combination or intersection is considered to be a state of the art alignment system. It has yet to be surpassed as the standard for alignment quality. It will be the baseline against which all novel models developed here will be compared.

2.3 Word-to-word Models of Translational Equivalence

The Candide models of translation present a number of variations on one possible method to use large corpora together with statistics in order to align parallel text. They are attractive because of their ability to encapsulate all relevant information inside a probabilistic

framework. However, this can also be seen as a weakness, preventing one from adding simple linguistic intuitions as factors external to the carefully designed probabilistic model. Melamed's body of work on models of translational equivalence, collected in [31], presents a well-motivated alternative to the Candide models. It represents the single largest influence on the novel work presented in this thesis. This preamble will serve to explain the two major intuitions behind Melamed's work, and then give an outline to the basic structure of his various methods for word alignment.

The first major factor contributing to the models presented in [31] was hinted at in Section 2.2.2, which described Candide Model 1. There, it was shown that when ignoring factors involving other links in an alignment, one could count hypothetical links by weighting the word co-occurrences in a sentence pair according to alignment probabilities. Equations 2.7 and 2.8 can be combined in order to create the following equation sequence for the initial EM iteration, where all translation probabilities for a word e are set to a uniform constant p:

$$\begin{aligned} P_{t_1}(f|e) &= z \cdot \sum_{s=1}^{|S|} \frac{P_{t_0}(f|e) \cdot cooc(e, f; E_s, F_s)}{\sum_{i=0}^{n_s} P_{t_0}(f|e_i)} \\ &= z \cdot \sum_{s=1}^{|S|} \frac{p \cdot cooc(e, f; E_s, F_s)}{p \cdot n_s} \\ &= z \cdot \sum_{s=1}^{|S|} \frac{cooc(e, f; E_s, F_s)}{n_s} \end{aligned}$$

where z is the normalizing factor that makes $P_{t_1}(f|e)$ a probability distribution.

As we can see from this re-writing, in the very first step of the model used to bootstrap all subsequent Candide models, the only information taken into account is the co-occurrence counts weighted by the inverse of the English sentence lengths. This makes a lot of sense, as words that co-occur more often are more likely to be translations, and those co-occurrences in shorter sentences should be given larger weight, as there is less chance that the French word is actually generated by some other English word. This starting point considers how often the two words in question occur together; however, it does not consider how often either one of the words that make up the pair occurs without the other word, or how often pairs that involve neither word occur. The power of EM allows Candide to build very good alignments, and from those, very good tables for $P_t(f|e)$, but one cannot help but wonder what would happen if a more sophisticated measure of relatedness was used as a starting point.

The idea of using a sophisticated correlation metric for word-alignment was explored

	f		$\neg f$	
e	$cooc(e, f)$	(a)	$cooc(e, \neg f)$	(b)
$\neg e$	$cooc(\neg e, f)$	(c)	$cooc(\neg e, \neg f)$	(d)

Table 2.1: A Co-occurrence Contingency Table

earlier in [14]. They used a metric called ϕ^2, which is related to the χ^2 distribution, in order to take advantage of all of the columns of the contingency table that relates the co-occurrences of two words. If we define $cooc(e, f)$ as follows:

$$cooc(e, f) = \sum_{s=1}^{|S|} cooc(e, f; E_s, F_s)$$

and we use $\neg w$ to represent any word but w, then the co-occurrence contingency table for a word pair is shown in Table 2.1. Given this contingency table, the ϕ^2 metric is defined to be:

$$\phi^2 = \frac{(ad - bc)^2}{(a + b)(a + c)(b + d)(c + d)} \tag{2.18}$$

Because of its connection to the χ^2 distribution, the ϕ^2 correlation metric makes some normality assumptions that do not always hold when dealing with text. In [11], Dunning suggested an alternative metric, a likelihood ratio that we will refer to as G^2. This method makes binomial assumptions in place of normality assumptions, and is generally considered to be better suited to working with text. It compares the distribution of f given that e is present (the first row of Table 2.1) to the distribution of f given that e is not present (the second row) to measure the degree of f's dependence on e. The formula for G^2 is as follows:

$$G^2 = -2 \log \frac{B(a; a + b, \frac{a}{a+b}) B(c; c + d, \frac{c}{c+d})}{B(a; a + b, \frac{a+c}{a+b+c+d}) B(c; c + d, \frac{a+c}{a+b+c+d})} \tag{2.19}$$

where $B(k; n, p)$ is the binomial probability of an event with probability p occurring k times in n samples; $B(k; n, p) = \binom{n}{k} p^k (1 - p)^{n-k}$.

In the cases of both ϕ^2 and G^2, the higher the score, the greater the correlation between the two words. Furthermore, both equations make use of all four cells of the contingency table, while Candide uses only cell a. Unfortunately, these scores are no longer probabilities, and cannot be used in EM as such. None the less, they remain statistically motivated. It will be shown later in this section how these scores can be used in an EM-like iterative procedure.

The other major idea explored in [31] is the notion of a **one-to-one constraint**. One of the greatest weaknesses of correlation metrics applied to word alignment is their inability to distinguish direct associations from indirect associations. Imagine a common phrase, such

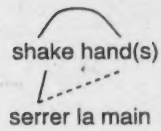


Figure 2.7: An Illustration of an Indirect Association

as, "shake hands," and its French translation "serrer la main," where *shake* corresponds to *serrer* and *hand* corresponds to *main*. The word *serrer* is not a direct translation of *shake*, it actually means to grip or to clasp. Because of this, *serrer* is unlikely to be used frequently in many corpora except in the context of shaking hands. On the other hand, *main* is the most frequently used translation for *hand*. Because *serrer* occurs most often in a sentence pair that contains both *shake* and *hand*, a correlation score for (*hand*, *serrer*) might be higher than the score for (*shake*, *serrer*). This high score for (*hand*, *serrer*) is an indirect association caused by the pair of direct associations between (*shake*, *hand*) and (*shake*, *serrer*). This is illustrated in Figure 2.7, where solid lines represent direct associations and dotted lines show indirect associations. Of course, there would be no confusion if we could disregard the possibility of the (*hand*, *serrer*) association because *hand* can be explained away with the much stronger association, (*hand*, *main*). This example shows that correlation scores are a starting point only. Indirect dependencies stop correlation metrics from being effective estimators of true translation likelihood. However, with a factor to take into account "explaining away," as it was alluded to above, a correlation metric could be used to create hypothetical links in real sentences, and then a new **link-based scoring metric** could be designed to make use of these link counts.

The problem of indirect associations is solved in [31] by making a one-to-one assumption. The assumption states that any given word in a sentence E can be linked to exactly one word in F, and vice versa, with the exception of the *null* word, which can link to multiple words. In many language pairs, this assumption is true more often than it is not; generally a single word in one language corresponds to a single word in another language. If this is not true, its restrictions may be alleviated by tokenizing one language with the other language in mind, so that some phrases are treated as words. A method for doing so is described in [29]. Although the one-to-one assumption eliminates the chance of making any many-to-one alignments, it also eliminates the need for anything resembling a fertility parameter. We will see later that alignment methods restricted by the one-to-one assumption can actually outperform other unrestricted methods.

With the one-to-one assumption in place, one can create an alignment very efficiently using a greedy algorithm. This algorithm links the two most similar words according to the scoring metric, removes them from their respective sentences, and then repeats the process until no words remain. This greedy process is generally referred to as **competitive linking** [31]. Competitive linking produces an approximation to A_{max}, the weighted maximum matching of the two sentences, where a weighting is defined as the sum of the link scores for all of the links in an alignment A connecting E and F:

$$A_{max} = \text{argmax}_{A \in \mathcal{A}} \sum_{l(i,j) \in A} score(e_i, f_j) \tag{2.20}$$

Links to *null* are counted every time a word is left unlinked, while co-occurrences with *null* are counted using the unigram frequency of the word. Competitive linking is substantially faster than the fastest maximum matching algorithm, and still achieves good results.

Having introduced the notions of competitive linking, correlation metrics, and link-based scoring metrics, we are now ready to sketch a skeleton for an alignment algorithm. Given any scoring function, $score(e, f)$ that relates two words one can refine scores using the following iterative process:

1. Initialize the *score* function according to co-occurrence counts and a correlation metric.
2. Use competitive linking with *score* to get link counts for all the sentences in the corpus.
3. Create a new *score* function according to link counts.
4. Repeat from step 2.

Melamed presented two methods for using hypothetical links to define a $score(e, f)$ function which measures the degree to which words e and f are related. One measure is purely probabilistic, while the other takes the form of a score from a statistical test. These will be described in the following subsections.

2.3.1 Probabilistic Model

The entire framework presented above and in [31] can be interpreted as a symmetric model of translation, given the correct scoring metric $score(e, f)$. Where the Candide models focused on the conditional probability:

$$P(F|E) = \sum_{A \in \mathcal{A}} P(F, A|E)$$

this work deals with the joint probability that the two sentences would be generated together:

$$P(F, E) = \sum_{A \in \mathcal{A}} P(F, A, E) \tag{2.21}$$

With this new equation, the focus is no longer on the act of translation: generating one sentence from an existing sentence. Instead, we view the problem as beginning with a single, hidden set of concepts to express. These concepts are then lexicalized simultaneously into the two languages E and F, which are linked according to their underlying concepts by the hidden alignment A.

Note that the definition of an alignment has changed to a similar, but different concept from the one used by Candide. In this new model, an alignment exists to indicate which words were generated from the same concept. In this way, an alignment indicates that two words are mutual translations of one another. Conversely, the definition according to Candide states that an alignment indicates which English words generated which French words. Unlike our new definition, the Candide relationship is not symmetrical, as a single link leading into an English word may or may not account for the entire English word, while a French word is always explained by a single link.

The above probability is further decomposed by assuming that a given sentence consists of a bag of words B, and an ordering over that bag, O. One can link the words in the bags for two parallel sentences according to an alignment A, and then order the words in both sentences according to the bags and the alignment. This results in the decomposition of:

$$P(F, A, E) = P(B_F, A, B_E) \cdot P(O_F, O_E | B_F, A, B_E) \tag{2.22}$$

With this decomposition, the words in the sentences have been separated from their order. The work done in [31] deals only with the first term from Equation 2.22: $P(B_F, A, B_E)$, treating all sentences as bags of words. If one later wished to use the bag-to-bag work in a complete sentence to sentence model, one would need to model $P(O_F, O_E | B_F, A, B_E)$ as well.

In order to reach a tractable algorithm, we must continue to decompose $P(B_F, A, B_E)$. We will do so by defining a generative story, in which the first step toward a (B_F, A, B_E) triple is deciding how many concepts will work together to generate B_F and B_E. We will call this term c, which also corresponds to the number of links $l(i, j)$ there are in the alignment A. There will be $c!$ ways to order all of the concepts in the generating bag. Each concept will generate either an (e, f) pair, or a word in one language and *null* in the other. Concepts generate their words independently of other concepts in the bag. Word pairs are

generated simultaneously based on the concept. We will refer to individual concepts using C, and the set of all possible concepts will be $\mathcal{C}$. This produces the following equation for $P(B_F, A, B_E)$:

$$P(B_F, A, B_E) = P(c) \cdot c! \cdot \prod_{l(i,j) \in A} \left[\sum_{C \in \mathcal{C}} P(e_i, f_j | C) \right] \tag{2.23}$$

If we assume that $|\mathcal{C}| = 1$ and that $P(c) \cdot c! = \mathbf{k}$ for some constant, $\mathbf{k}$, Equation 2.23 simplifies to:

$$P(B_F, A, B_E) = \mathbf{k} \cdot \prod_{l(i,j) \in A} P(e_i, f_j) \tag{2.24}$$

As $\mathbf{k}$ is constant, to find the most likely alignment for two bags of words, we want to find:

$$\begin{aligned} \text{argmax}_{A \in \mathcal{A}} P(B_F, A, B_E) &= \text{argmax}_{A \in \mathcal{A}} \prod_{l(i,j) \in A} P(e_i, f_j) \\ &= \text{argmax}_{A \in \mathcal{A}} \sum_{l(i,j) \in A} \log P(e_i, f_j) \end{aligned} \tag{2.25}$$

If we let $score(e_i, f_j) = \log P(e_i, f_j)$ then this equation becomes identical to Equation 2.20, which is the objective function for competitive linking. One can estimate $P(e, f)$ values by counting links directly from a sample of alignment space. In competitive linking, the sample consists of only the current maximum likelihood estimate of A_{max} for each sentence in the corpus. If we define $links(e, f)$ to be the number of times e and f are linked in the entire corpus according to our latest round of competitive linking, then $P(e, f)$ can be calculated as the chance that any concept generates the pair (e, f) as opposed to any other possible pair:

$$P(e, f) = \frac{link(e, f)}{\sum_{(e', f')} links(e', f')} \tag{2.26}$$

The final result is a scoring metric of the form $score(e, f) = \log P(e, f)$. When this scoring function is plugged into the algorithm described above, the process can iteratively improve the function as it improves the alignments it produces. Under the assumptions defined above, this finds an approximation to the maximum likelihood value of A_{max} for each sentence pair in our corpus. Using either of the correlation metrics discussed above serves as a method to set a good starting point for the search. The scoring metric in Equation 2.26 is generally not used in practice, it serves primarily as a justification for the entire algorithm for the probabilistically minded. This justification is important, though, as we will use some concepts developed here in the novel algorithm presented later in the thesis. However, the explicit noise model to be described in the following section provides a much more powerful word-to-word scoring method.

2.3.2 Explicit Noise Model

A naïve way to make a refined $score(e, f)$ function based on link counts is to use link counts alone, which is what is done in the probabilistic model. Doing so does not account for a potentially important source of information, though, as we still have access to the co-occurrence counts that were used to create the initial correlation-based scoring metric. The explicit noise model, also explained in [31], looks at the ratio of the number of times a word pair is linked compared to the number of times it co-occurs in order to decide on the score for the pair. Two words that are linked nearly every time they co-occur in a sample are intuitively stronger translations than two words that are linked only half of the time they co-occur. This notion regarding co-occurrences and hypothetical links will become central to the novel word alignment model presented later in this thesis.

The explicit noise model parameterizes the noise present in a given hypothetical alignment created by competitive linking. It models the probability λ^+ of a link occurring between two words that are true mutual translations, and the probability λ^- of a link occurring for two words that are not mutual translations. It uses these two probabilities as parameters in a scoring metric to measure the strength of the connection between two word types. It is important to note that whether or not a translation is true is decided by word type, not token. We will use these two parameters in a mixture model, so no one pair of types need belong entirely to λ^+ or λ^-.

To motivate the method required to learn these parameters, we will first describe the scoring metric they would allow us to create. If one knew an accurate value for both λ^+ and λ^-, one could use a likelihood ratio to determine how strongly a given type pair (e, f) fit with either parameter. If we observed $cooc(e, f)$ co-occurrences of e and f, and we observed $links(e, f)$ links between them, it is like making $cooc(e, f)$ tests, $links(e, f)$ of which are positive. In other words, the probability of linking a word pair $links(e, f)$ times out of $cooc(e, f)$ opportunities can be modeled by a binomial distribution. The ratio between the probability of this series of events according to λ^+ and the probability according to λ^- will determine our level of certainty as to whether or not this series of tests was generated by λ^+. If λ^+ is more likely, then the two words should probably be linked. If λ^- is more likely, then the words are likely noise and should not be linked. The log of this ratio provides us with the following scoring metric:

$$score(e, f) = \log \frac{B(links(e, f); cooc(e, f), \lambda^+)}{B(links(e, f); cooc(e, f), \lambda^-)} \tag{2.27}$$

When this score is positive, it indicates that the types fit the mutual translation model better

than the false translation model. A negative score indicates that the types are more likely to not be translations. A score of greater magnitude in either direction indicates that the metric is more certain of its predication.

The task remains to estimate the parameters λ^+ and λ^- for a given alignment of the corpus. Fortunately, there is a maximum likelihood method that allows us to achieve this goal. Let ρ be the probability that a given pair of English, French types is a true mutual translation. Then we can assign a probability to our current set of links according to our current noise model with the following product over all co-occurring type pairs:

$$P_{\lambda^+,\lambda^-}(\mathit{links}) = \prod_{(e,f)} \rho B(\mathit{links}(e,f); \mathit{cooc}(e,f), \lambda^+) \\ +(1-\rho)B(\mathit{links}(e,f); \mathit{cooc}(e,f), \lambda^-) \tag{2.28}$$

This is not the probability that our current set of links are correct. This is the probability that our noise model assigns to our current set of links. The higher the probability assigned, the closer our parameters come to actually modeling the noise present in the hypothetical links. The degrees of freedom in the model can be decreased by stating ρ in terms of λ^+ and λ^-. If we let λ be the probability that a link occurs given a co-occurrence, then we can say that

$$\lambda = \rho\lambda^+ + (1-\rho)\lambda^- \tag{2.29}$$

We can get λ from our current set of links; it is the total number of links divided by the total number of co-occurrences. With this term and Equation 2.29, we can express ρ in terms of the other two variables, allowing us to hill-climb over a two-dimensional space, searching for a (λ^+, λ^-) pair that maximizes the probability defined in Equation 2.28. The parameter space is fairly well-behaved, if one uses an appropriately large step size, the search should not stall in local maxima.

To create a more precise noise model, one can use different λ parameters for different categories of words. One can conditionalize on any number of features of words, including part of speech or frequency of co-occurrence. The counts of links and co-occurrences are then divided up according to these features, and the parameters are optimized separately. This conditional noise model has been shown in [31] to increase accuracy in some cases.

2.3.3 Word-to-word Model Summary

The word-to-word models of translation are very different animals from the various Candide models. These methods focus on developing accurate and meaningful scores for word

types. They generally assign the same value to any given pair of word types, the conditional method mentioned above being the exception to this rule. These models depend on the one-to-one assumption to allow them to create accurate alignments on a bag-to-bag level. Meanwhile, Candide operates on a sequence-to-sequence level. The wealth of different tables available to Candide's more advanced models allows it to assign a vast array of scores to a given pair of word types, depending on their sentence positions and the other links in the alignment. Because of these differences, and because the current standard for alignment scoring had not yet been developed as the word-to-word models were being published, there exists no direct comparison in the literature between the accuracy of these two methods. However, the importance of the word-to-word models does not lie in their role as a competitor to Candide. The simplicity and the modularity of the word-to-word models presented in this section make them very valuable. We will borrow both inspiration and algorithms from these methods as we develop the novel alignment algorithm presented in this thesis.

2.4 Maximum Entropy

Maximum entropy modeling provides a general-purpose technique for modeling a probability distribution. Of particular interest to natural language researchers is conditional maximum entropy, which is a supervised learning method that allows one to model a probability distribution $P(c|x)$ over a set of possible classes C given an example drawn from a set of possible examples X. This section will discuss conditional maximum entropy, and its application to probabilistic translation models such as the Candide models. The brief introduction to maximum entropy that follows draws primarily from the explanations found in [3] and [35].

The main principle of maximum entropy methods is to model what is known about a sample distribution, and to assume uniformity, or maximum entropy, over what is not known. For example, imagine that we are attempting to model the Candide translation probability $P_t(f|e)$ for the English word $e = \textit{computer}$. Since our model is exclusively built for a specific e, we will alter notation to indicate that the conditionalizing e is built into the model, and refer to translation probability as $P_{t_e}(f)$. If all we were told was that there were exactly three translations for *computer*: *ordinateur*, *informatique*, and *calculateur*, we could find the maximum entropy distribution for $P_{t_e}(f)$ subject to the constraint that:

$$P_{t_e}(\textit{ordinateur}) + P_{t_e}(\textit{informatique}) + P_{t_e}(\textit{calculateur}) = 1$$

The maximum entropy principle would in turn indicate that we had to give each translation uniform weight:

$$P_{t_e}(ordinateur) = P_{t_e}(informatique) = P_{t_e}(calculateur) = \frac{1}{3}$$

If we know more, then maximum entropy modeling allows us to take advantage of additional knowledge while still maintaining uniformity in our distribution when we can. For example, if we also knew that *computer* was translated as *ordinateur* 80% of the time, we could find the maximum entropy distribution subject to the constraints:

$$P_{t_e}(ordinateur) = 0.8$$
$$P_{t_e}(ordinateur) + P_{t_e}(informatique) + P_{t_e}(calculateur) = 1$$

which would produce the following distribution:

$$P_{t_e}(ordinateur) = \frac{4}{5}; \quad P_{t_e}(informatique) = P_{t_e}(calculateur) = \frac{1}{10}$$

With these examples, it has been relatively clear what the maximum entropy distribution given the constraints should be. The maximum entropy framework provides us with an algorithmic method to find the most uniform distribution that agrees with any set of constraints that do not contradict each other. It will also provides us with a mechanism to conditionalize on any number of features that may be present in a given example. The nature of the features and the resulting model will be described in the following two sections.

2.4.1 Features in Maximum Entropy

We will refer to the aspects of the sample that we intend to model as features. As we formalize this intuition, the notion of a feature will be defined in such a way so as to make the types of features available as versatile as possible. In conditional maximum entropy, a feature is a function of an example and a class. We will use $h_k(x, c)$ to represent the k^{th} feature function of our model, where $x \in X$ is an example from our sample, and $c \in C$ is a class that an example may or may not fall into. We will consider our feature functions to be binary functions, returning 1 if the feature is present, and 0 if it is not. In practice, most algorithms will allow for features that return any non-negative value.

Suppose we wanted to improve our model for $P_{t_e}(f)$ where $e = computer$ by conditionalizing on the English sentence s that *computer* appears in, creating $P_{t_e}(f|s)$. In this case, the resulting French word f is equivalent to our class c, and the sentence s containing *computer* is our example x. There are many aspects of the example that we could specify to trigger our feature functions: the grammatical relationships e partakes in, or the

part-of-speech e was labeled with by an independent part-of-speech tagger, or we could look at the other words in the x. But we would not want to make a feature especially for x = "I am taking my computer in to get it fixed on Monday," unless that sentence occurs so frequently in our corpus that memorizing the translation of e for that sentence would significantly increase our accuracy. Consequently, there is usually a sub-function within h that acts on an aspect of x, so that h will return 1 for many different examples, given the correct class. Some example features that might be used in our translation domain are listed below:

$$h_1(x,c) = \begin{cases} 1 \text{ for any } x, \text{ if c=}\textit{calculateur} \\ 0 \text{ otherwise} \end{cases}$$

$$h_2(x,c) = \begin{cases} 1 \text{ if } \textit{book} \text{ follows } \textit{computer} \text{ in } x, \text{ and c=}\textit{informatique} \\ 0 \text{ otherwise} \end{cases}$$

$$h_3(x,c) = \begin{cases} 1 \text{ if } \textit{computer} \text{ appears in } x \text{ as an object of a verb and c=}\textit{ordinateur} \\ 0 \text{ otherwise} \end{cases}$$

In the examples above, h_1 would be useful for modeling the context-free probability of translating *computer* into *calculateur*. h_2 and h_3 serve as counters for how often a particular translation happens in a particular context. Generally, a large list of possible features is generated using feature templates [3]. A smaller number of features are then selected to be modeled. Features can be selected according to their frequency [35] in the sample or some other more complicated feature selection mechanism [3, 38].

Features affect the maximum entropy modeling process by imposing constraints on the model. Specifically, we say that for every feature, the expected feature count in our sample given our current model must match the actual feature count in our sample. Let S represent our labeled sample, with class labels $c(x)$ for all $x \in S$ and let $P(c|x)$ represent our maximum entropy conditional model, then our constraints on our model are as follows:

$$\forall k : \frac{1}{|S|} \sum_{x \in S} h_k(x, c(x)) = \frac{1}{|S|} \sum_{x \in S} \left[\sum_{c \in C} P(c|x) h_k(x,c) \right] \tag{2.30}$$

In other words, the model must be accurate enough on the sample to label the examples in such a way that the feature counts remain intact. The following section will briefly outline the search for a $P(C|X)$ that obeys these constraints but also obeys the principle of maximum entropy when left unconstrained.

2.4.2 Modeling the Distribution

With constraints of the form listed above, it can be shown [38] that a single distribution P that maximizes entropy exists. Furthermore, this distribution will have an exponential form:

$$P(c|x) = \frac{1}{Z(x)} \exp\left(\sum_k \lambda_k h_k(x,c)\right) \tag{2.31}$$

where Z is a normalizing term to ensure a valid probability distribution over all possible c:

$$Z(x) = \sum_{c \in C} \exp\left(\sum_k \lambda_k h_k(x,c)\right) \tag{2.32}$$

The λ terms in these equations represent weights given to features. Given a set of features, it is the set of λ weights corresponding to those features that defines the distribution of a maximum entropy model. Therefore, the training task is a constrained search for the set of λ parameters that maximize entropy, where entropy can be defined mathematically as:

$$H(P) = -\frac{1}{|S|} \sum_{x \in S} \sum_{c \in C} P(c|x) \log P(c|x)$$

It has been shown in [38] that for any set of constraints following the format in Section 2.4.1, a unique set of weights will maximize entropy. As is explained in [3], this constrained optimization problem can be converted into a dual unconstrained optimization problem using Lagrange multipliers. This dual space is very well-behaved, it is convex and has a unique maximum. One can search the dual space using any number of numerical methods, including maximum entropy-specific searches such as generalized iterative scaling [8] and improved iterative scaling [38], or more general purpose methods, such as gradient-based method or variable metric methods [24]. A comparison of several methods in terms of efficiency can be found in [24]. The details of the calculus and the numerical methods are unimportant to the work presented here. What is important is that such a parameter set can be found reliably.

Before moving on to a description of how maximum entropy has been used to improve Candide, there are some points about the model produced by maximum entropy that are not clear from the mathematical description. Despite the fact that a conditional maximum entropy model possesses exactly one parameter for each feature, it discriminately models the conditional distribution, allowing it to account for some of the relationships between features. Consequently, if two features are strong indicators of a specific class, but always appear together in the sample, then they will each be given half weight. In this way, maximum entropy does not double-count features in the same way as Naïve Bayes [35].

However, this also means that if a set of γ overlapping features can uniquely identify an example x from the sample, then the model will memorize the class of x exactly. Presented with a new example x' that also triggers the same γ features, it will return near-absolute certainty that x' is of the same class as x, that is $P(c(x)|x') \approx 1$. This level of certainty is present despite having seen only one example that matches all γ features. In this way, maximum entropy is prone to over-fitting. Smoothing methods exist to overcome this problem with sparse data. A simple and effective smoothing method for the improved iterative scaling algorithm is presented in [6].

2.4.3 Maximum Entropy to Enhance Candide

With the examples given above, it should come as no surprise that maximum entropy can be used to improve the various distributions present in the Candide models. Because of its versatile feature definition, maximum entropy provides us with tools to conditionalize on any number of interesting events in sentences, without deriving a new model, or conditionalizing on all similar events. For example, take the case where our feature selection algorithm has found that checking if the word following *computer* is *monitor* gives information on the translation of *computer*. Furthermore, it has determined that there are enough relevant examples present in the sample to support the estimation of such a feature. In this case, conditional maximum entropy allows us to store a probability for $P_{t_e}(f|e_{i+1} = \textit{monitor})$ without necessarily storing a probability for $P_{t_e}(f|e_{i+1} = e')$ for all possible e'. In this way, other e' without enough data in the sample need not be modeled in the same manner.

In [3], maximum entropy is used to create further conditionalized versions of Candide's translation model P_t. They do this by training a separate $P_{t_e}(f|s)$ model for each possible e in the English vocabulary, where s is the sentence in which e appears. Since maximum entropy is a supervised training method, these models are trained by using the current Viterbi alignments from Candide as the correct class labels. Every time e is linked to a French word f, the sentence in which the link occurred is entered into P_{t_e}'s maximum entropy training set with class label f. Features are extracted from examples according to feature templates. In [3], five templates are used. These are summarized in Table 2.2. In this table the symbols $\square$ and $\diamond$ are place holders that can represent any English and French word respectively. So, the example feature function h_2 from Section 2.4.1 would be generated by the template $(t = 2, \diamond = \textit{informatique}, \square = \textit{book})$. Feature template $t = 1$ is used to model the unconditionalized probability of translating to a French word. If only $t = 1$ templates were used, maximum entropy would produce the exact same probability table as

t	$h_{t,\diamond,\square}(x,c) = 1$ if and only if ...
1	$c = \diamond$
2	$c = \diamond$ and $\square$ is in the position after e
3	$c = \diamond$ and $\square$ is within 3 words after e
4	$c = \diamond$ and $\square$ is in the position before e
5	$c = \diamond$ and $\square$ is within 3 words before e

Table 2.2: An Example of Maximum Entropy Feature Templates

the original $P_t(f|e)$ from Candide.

These templates generate a very large number of potential features. The feature selection algorithm described in [3] then picks out those features that are informative and can be estimated reliably from the sample. Once a model for $P_{t_e}(f|s)$ is trained using maximum entropy and the current Viterbi alignment, it is then used in place of $P_t(f|e)$ inside the appropriate Candide model. [3] shows anecdotal evidence to indicate that this modeling does improve translation quality. No experiments were conducted to see if alignment quality also improved. The notion of conditionalizing on a potentially variable number of features in order to improve the accuracy of a probability model will be used in the novel model explained later in this thesis.

2.5 Syntax in Translation

All of the alignment methods described thus far have treated sentences as either bags or sequences of words. However, human languages have a much richer structure than linear sequences. In language, words can be connected to one another by specific syntactic relationships, forming a tree of connected phrases and ideas. Armed with a parser designed to process the appropriate natural language, one can automatically derive a possible syntactic structure for a sentence from its linear sequence. These parsers can be quite accurate, so much so that they can be safely used as a component of another system. It is hypothesized that syntactic structure is mostly conserved during translation. Syntactic methods in machine translation attempt to use syntactic structure to constrain or guide word alignment algorithms. This is intended to strengthen the word ordering component of translation models.

Syntactic structures decompose word sequences into substructures in a hierarchical or tree-like manner. These substructures are often referred to as phrases. A parse of a sentence would treat the entire sentence as one phrase, which in turn is divided into phrases, which are divided into phrases of their own. At the leaf nodes of the tree, every word is its own

Linear Format:

```
The reboot causes the host to discover all the devices.
```

Parsed Format:

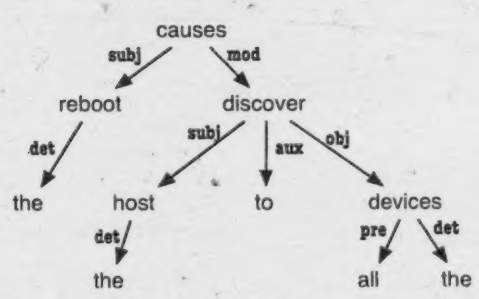


Figure 2.8: A Sentence Parsed Using a Dependency Grammar

singleton phrase. The nature of this hierarchy is defined by the structure of the grammar used to parse the sentence. We will focus our discussion on dependency grammars. An example of a sentence parsed using a dependency grammar is shown in Figure 2.8. All nodes in a dependency tree are labeled with a word from the parsed sentence, and each word is represented by exactly one node. The tree is interpreted by saying that child nodes modify their parents in some grammatical sense. For example, in the right-most subtree in Figure 2.8 the word *devices* is modified by both *all* and *the*, as they provide extra information about the primary concept, *devices*. The labels on the edges indicate the nature of the modification. In our example sentence, the word *host* is modified by its determiner, *the*; by the same token, *host* modifies *discover* by providing a verb subject.

A word is said to dominate all of the words that descend from it in the tree. The span over these words in the original sentence also defines a phrase with the root of the subtree as a head word. The phrases defined by the dependency tree in Figure 2.8 include, "the reboot," headed by *reboot*, "all the devices," headed by *devices*, "the host to discover all the devices," headed by *discover*, and the original sentence itself, headed by *causes*. Dependency trees are generally constrained to be connected and planar. If one were to draw the dependency edges over the words as they appear in their original linear sequence, then no edges should cross one another [22]. This reflects the natural tendency in language to complete one thought before moving on to another.

Some alignment methods, such as [2] and [22] impose structural constraints through a synchronous parsing procedure. In these alignment algorithms, the two sentences to be

aligned are modeled as being generated at the same time; that is, a word and its translation are simultaneously generated according to some grammar. This dual generation process implies an alignment, as well as a dependency tree over both sentences. Because of the nature of the generative process, the two sentences must be given isomorphic tree structures. The cost for aligning a pair of words is determined using the ϕ^2 correlation metric mentioned earlier. A search conducted using dynamic programming can find the optimal alignment and structure in a tractable amount of time. In [2], the structure is allowed to take whatever form best fits the sentence pair. In [22], one of the two sentences is parsed in advance by an external natural language parser. They then use the same synchronous parsing process as in [2] to align the sentences, but the process is constrained so that the structure inferred by the algorithm matches that of the input parse. This method was designed specifically for use in an alignment task. Though this process did increase the quality of the syntactic structures that were induced in the unparsed sentences by the alignment, the alignments themselves were not able to match GIZA++ in alignment accuracy.

Many other methods which use structure in translation and alignment exist. Both [42] and [32] are similar to the work in [2]. They provide alternative grammatical formalisms and dynamic programming frameworks to simultaneously infer structure and alignment in sentence pairs. Other methods, such as [43], use a probability model over operations on parse trees. These methods model the transformation of a parse tree in the target language into a tree in the source language. They do so using the same sorts of techniques used to model sequence-to-sequence translation in Candide.

It was pointed out in [12] that all of the syntax-based alignment methods are attempting, in one form or another, to leverage the notion of phrasal cohesion. Phrasal cohesion is the hypothesis that when a syntactic phrase is translated, its composite words will tend stay together in the translation. Words within a phrase can be re-ordered, and phrases themselves can be re-ordered, but two distinct phrases in the source language should not overlap one another in the translation. A study conducted in [12] indicates that dependency structures maintain the strongest degree of cohesion during translation, when compared to other natural language structures such as constituency trees. In this thesis, we will use the notion of cohesion as an auxiliary tool. Its role will be to constrain the alignments generated by the novel probability model that is central to this thesis.

Chapter 3

Probability Model

The centerpiece of this thesis is a new alignment probability model [7] designed to be used as a component of a complete alignment system. This model attempts to capture the strengths of both Candide and Melamed's work in a relatively simple framework. We also incorporate flexible features as is done with the maximum entropy extensions to Candide. This chapter will begin by motivating and explaining this new alignment model and its parameters. These explanations will be supplemented by an example of the model in action.

3.1 Derivation

In [31], two models are presented to estimate the similarity between word types across languages. These models are trained based on a corpus labeled with potentially noisy alignments. The probabilistic model uses only link counts, and performed fairly poorly. The explicit noise model incorporates co-occurrence counts as well as link counts into a scoring metric that attempts to determine true correspondences from false ones. According to the experiments conducted in [31], this second method faired much better. In this section, we derive a model that is more like Candide than the explicit noise model, but unlike Candide, takes the ratio of links to co-occurrences into account.

Like Candide, we view sentences as ordered sequences of words: $E = e_1, e_2, \ldots, e_n$ and $F = f_1, f_2, \ldots, f_m$. To prevent confusion, we will use $\overline{e}_i$ to represent the type of the i^{th} word in the sequence E, and e_i to represent the specific token that is the i^{th} word. The token contains information on the i^{th} word's type as well as its position in the sentence. We will use a similar notation for the French types ($\overline{f}_j$) and tokens (f_j). As we have seen from the related work, the exact definition of an alignment can vary from model to model. For the purposes of this model, an alignment A is defined as a sequence of t **links**, $\{l_1, l_2, \ldots, l_t\}$. We will refer to consecutive subsequences of A as $l_x^y = \{l_x, l_{i+1}, \ldots, l_y\}$. Each link l_k

is an English-French token pair such that a link $l_k = l(e_i, f_j)$ exists if and only if e_i and f_j are a translation (or part of a translation) of one another in this particular sentence pair. We define the **null link** $l(e_i, f_0)$ to exist if e_i does not correspond to a translation of any French word in F. The null link $l(e_0, f_j)$ is defined similarly. This definition of a link as a mutual translation indicator is similar to the definition used in the probabilistic model of translational equivalence (see Section 2.3.1). We place the following restrictions on the links in an alignment: every token in E and F must participate in at least one link, and any token linked to e_0 or f_0 must not participate in any other links. In this way, the presence of each token in the sentence pair is explained in the context of its translation. We will impose further restrictions in the alignment algorithm built on top of this model, but the model itself does not require them.

We define the alignment problem as the task of finding the most likely alignment for the given sentences. Mathematically, we are looking for a link sequence A that maximizes $P(A|E,F)$ for a given English-French sentence pair (E,F). The Candide models provide values for $P(F,A|E)$, which can in turn be used to calculate $P(A|E,F)$ according to Equation 2.2. Because this equation calculates $P(A|E,F)$ as normalized $P(F,A|E)$ values, an alignment that maximizes $P(F,A|E)$ will also maximize $P(A|E,F)$. The model derived in this section will model $P(A|E,F)$ directly, making Equation 2.2 unnecessary. In the Candide models of translation, alignments exist as artifacts of which English words generated which French words. This new model does not state that one sentence generates the other. Instead it takes both sentences as given, and uses the sentences to determine an alignment. The central equation to this model is as follows:

$$P(A|E,F) = P(l_1^t|E,F) = \prod_{k=1}^{t} P(l_k|E,F,l_1^{k-1}) \tag{3.1}$$

Please note that we have not yet made any simplifying assumptions in our decomposition. Generating the alignment is a sequence of decisions, where each decision places a single link. The probability of each decision is conditionalized on all of the information available about the sentences (E and F) as well as all of the decisions made up to the current point (l_1^{k-1}). With this total information decomposition, the order in which decisions are made does not matter.

However, this total information model is not practical. To make computation feasible and to promote generalization, we will have to make simplifying assumptions that remove some information from our conditionalizing terms. Before we do so, we will re-write $P(l_k|E,F,l_1^{k-1})$ in order to draw out terms that we can reason about more easily. Let

$C_k = \{E, F, l_1^{k-1}\}$ represent all of the information available as the generative process places the k^{th} link. We will refer to this as the context of l_k. The context C_k contains all information about both sentences including all tokens, while a link $l_k = l(e_{i_k}, f_{j_k})$ states that the token e_{i_k} was linked to the token f_{j_k}. Therefore, both C_k and l_k imply that $\overline{e}_{i_k}$ and $\overline{f}_{j_k}$ co-occurred in the sentence pair. With this in mind, we can rewrite $P(l_k|C_k)$ as follows:

$$\begin{aligned} P(l_k|C_k) &= \frac{P(l_k, C_k)}{P(C_k)} = \frac{P(C_k|l_k)P(l_k)}{P(C_k, \overline{e}_{i_k}, \overline{f}_{j_k})} \\ &= \frac{P(C_k|l_k)}{P(C_k|\overline{e}_{i_k}, \overline{f}_{j_k})} \cdot \frac{P(l_k, \overline{e}_{i_k}, \overline{f}_{j_k})}{P(\overline{e}_{i_k}, \overline{f}_{j_k})} \\ &= P(l_k|\overline{e}_{i_k}, \overline{f}_{j_k}) \cdot \frac{P(C_k|l_k)}{P(C_k|\overline{e}_{i_k}, \overline{f}_{j_k})} \end{aligned} \tag{3.2}$$

Here $P(l_k|\overline{e}_{i_k}, \overline{f}_{j_k})$ is the probability of a link given the types of the two words to be linked. This can also be viewed as the probability of linking any particular co-occurrence of the two types. We will refer to this term as **link probability**. The context ratio $\frac{P(C_k|l_k)}{P(C_k|\overline{e}_{i_k}, \overline{f}_{j_k})}$ modifies the link probability, providing context-sensitive information regarding the sentences and previous link decisions.

This re-write has not introduced any simplifying assumptions into our derivation; it has only served to rearrange terms into a more manageable format. $C_k = \{E, F, l_1^{k-1}\}$ remains too complex to estimate the probabilities required for our context ratio directly. Instead of working with C_k directly, we will consider only a subset of the relevant features from a link's context. Suppose H_k is a set of context-related features such that $P(l_k|C_k)$ can be approximated by $P(l_k|\overline{e}_{i_k}, \overline{f}_{j_k}, H_k)$. Let $C'_k = \{\overline{e}_{i_k}, \overline{f}_{j_k}\} \cup H_k$. $P(l_k|C'_k)$ can then be decomposed using the same derivation as above.

$$\begin{aligned} P(l_k|C'_k) &= P(l_k|\overline{e}_{i_k}, \overline{f}_{j_k}) \cdot \frac{P(C'_k|l_k)}{P(C'_k|\overline{e}_{i_k}, \overline{f}_{j_k})} \\ &= P(l_k|\overline{e}_{i_k}, \overline{f}_{j_k}) \cdot \frac{P(H_k|l_k)}{P(H_k|\overline{e}_{i_k}, \overline{f}_{j_k})} \end{aligned} \tag{3.3}$$

In the second line of this derivation, we can drop $\overline{e}_{i_k}$ and $\overline{f}_{j_k}$ from C'_k, leaving only H_k, because they are implied by the events which the probabilities are conditionalized on. We are approaching a point where all of the terms in this equation have an intuitive meaning. One further simplification must be dealt with before we can address feature set probabilities, though. The meaning of $P(H_k|l_k)$ is unclear, as $l_k = l(e_{i_k}, f_{j_k})$ is a relationship between two specific tokens. Since tokens are unique by definition, the probability of a particular feature set given a link between tokens should be either zero or one. This extremism will

limit generalization. To prevent this, we will approximate this value using feature probability given any link between the types of the two tokens. This **type link** will be denoted by $\bar{l}_k = l(\overline{e}_{i_k}, \overline{f}_{j_k})$. This creates the approximation:

$$P(l_k|C_k) \approx P(l_k|C'_k) \approx P(l_k|\overline{e}_{i_k}, \overline{f}_{j_k}) \cdot \frac{P(H_k|\bar{l}_k)}{P(H_k|\overline{e}_{i_k}, \overline{f}_{j_k})} \tag{3.4}$$

We are left only with the task of approximating $P(H_k|\bar{l}_k)$ and $P(H_k|\overline{e}_{i_k}, \overline{f}_{j_k})$. To do so, we will assume that all context features $h \in H_k$ are conditionally independent given either $\bar{l}_k$ or $(\overline{e}_{i_k}, \overline{f}_{j_k})$. Although this assumption does not hold for linguistic data, it has been shown to create effective learning tools in other areas such as text classification [34]. This assumption allows us to do a Naïve Bayes-like product of conditional probabilities to estimate both $P(H|\bar{l}_k)$ and $P(H|\overline{e}_{i_k}, \overline{f}_{j_k})$. Our final approximation to alignment probability $P(A|E,F)$ is as follows:

$$P(A|E,F) \approx \prod_{k=1}^{t} \left(P(l_k|\overline{e}_{i_k}, \overline{f}_{j_k}) \cdot \prod_{h \in H_k} \frac{P(h|\bar{l}_k)}{P(h|\overline{e}_{i_k}, \overline{f}_{j_k})} \right) \tag{3.5}$$

In the context of each link in the alignment, only a few features will be active. The inner product is understood to be only over those features h that are present in the current context. This approximation will cause $P(A|E,F)$ to no longer be a well-behaved probability distribution, though as in Naïve Bayes classifiers, it can potentially be an excellent estimator for the purpose of ranking alignments.

3.2 Parameter Estimation

This model was developed with the primary goal of improving an existing word alignment. Therefore, use of a method like EM to set parameters and induce alignments in an unaligned corpus has not yet been investigated. For the purposes of this thesis we will assume that we are given an initial training corpus with established, but potentially noisy alignments. Given an aligned training corpus, the probabilities needed to calculate Equation 3.5 can be obtained directly.

In order to describe parameter estimation, we will adopt a notation that uses $|event|$ to indicate the total number of times a given event has been observed throughout the aligned parallel corpus. To compare with the notation used in the related work section, the following equalities hold:

$$|l(\overline{e}_{i_k}, \overline{f}_{j_k})| = link(\overline{e}_{i_k}, \overline{f}_{j_k})$$
$$|\overline{e}_{i_k}, \overline{f}_{j_k}| = cooc(\overline{e}_{i_k}, \overline{f}_{j_k})$$

Given link counts and co-occurrence counts, the link probability term from Equation 3.5 can be calculated using the ratio of links to co-occurrences:

$$P(l_k|\overline{e}_{i_l}, \overline{f}_{j_k}) = \frac{|l(\overline{e}_{i_k}, \overline{f}_{j_k})|}{|\overline{e}_{i_k}, \overline{f}_{j_k}|} \tag{3.6}$$

The feature probabilities required for the context ratio can be calculated similarly. For any co-occurring pair of words (e_{i_k}, f_{j_k}), we check whether the feature h occurs within the context of the co-occurrence. If it does, then the count of $|h, \overline{e}_{i_k}, \overline{f}_{j_k}|$ is incremented. If this co-occurring word pair is also linked in our training corpus, then the count of $|h, l(\overline{e}_{i_k}, \overline{f}_{j_k})|$ is incremented as well. Feature probabilities are then calculated using the following two equations:

$$P(h|\overline{l}_k) = \frac{|h, l(\overline{e}_{i_k}, \overline{f}_{j_k})|}{|l(\overline{e}_{i_k}, \overline{f}_{j_k})|} \tag{3.7}$$

$$P(h|\overline{e}_{i_k}, \overline{f}_{j_k}) = \frac{|h, \overline{e}_{i_k}, \overline{f}_{j_k}|}{|\overline{e}_{i_k}, \overline{f}_{j_k}|} \tag{3.8}$$

Please note that features under this model are defined solely in terms of aspects of a link's context. Unlike maximum entropy, there is no class parameter attached to features.

In the derivation of this model, it was establish that when generating the k^{th} link, one must consider all of the links added to the link sequence up to this point, that is $\{l_1 \ldots l_{k-1}\}$. Using the total information model described in Equation 3.1, the order in which links are added to a link sequence does not matter, as the probabilities used there are abstract and ideal. The simplified, tractable model of $P(A|E,F)$ shown in Equation 3.5 defines H_k as a subset of features present in the context of the k^{th} link. This allows for features that depend on previous links. However, since features need not be symmetric, it is quite possible that Equation 3.5 will assign different probabilities to different orderings of the same sequence of links. Furthermore, in a training corpus all links are present simultaneously, with no indication of the order in which they were established. If, during training, one sees a link $l_{k'}$ that would cause a feature to be active for l_k, how does one know whether or not $l_{k'}$ was really placed before l_k, or if it will be placed before l_k when using the model to build new alignments?

To address these concerns, one must impose a deterministic ordering on the links in A when determining whether or not a feature is present in a given context. This ordering can be arbitrary, so long as the same ordering is used in both training and probability evaluation. In this way, one can be certain that two sequences containing identical links will always be given the same probability. This also ensures that a feature that was important

during training will remain important when the model is used to build alignments. A simple solution to the ordering problem would be to order links according the position of their French tokens in the French sentence, breaking ties with the English token positions. In the algorithm presented later in this thesis, links are ordered according to the context-free link probability $P(l_k|\overline{e}_{i_k}, \overline{f}_{j_k})$ of the two types involved. This was selected because it has an intuitive appeal of allowing more certain links to provide context for others.

Once collected, parameters can be stored in two tables. The first table stores link probabilities $P(l_k|\overline{e}_{i_k}, \overline{f}_{j_k})$. It has an entry for every word pair that was linked at least once in the training corpus. Its size is the same as the translation table in the Candide models. The second table stores feature probabilities, $P(h|\overline{l}_k)$ and $P(h|\overline{e}_{i_k}, \overline{f}_{j_k})$. For every linked word pair, this table has two entries for each active feature. If we let $\mathcal{H}$, $\mathcal{E}$ and $\mathcal{F}$ be the sets of all possible features, English words and French words respectively, then in the worst case this table will be of size $2 \times |\mathcal{H}| \times |\mathcal{E}| \times |\mathcal{F}|$. In practice, it is much smaller as most contexts activate only a small number of features. The features used in our implementation of this model are described in Section 4.2.1.

3.3 An Illustrative Example

This section will present an example of parameter estimation and probability evaluation using this new $P(A|E,F)$ model. We will get parameter values from an aligned corpus consisting of a single sentence pair, and the trained model will then be used to evaluate the sentence it was trained on as well as a new alignment. Figure 3.1 shows the single sentence pair that will be used as our aligned corpus. To further simplify matters, we will concern ourselves with only one feature. This feature h is active for an (e_{i_k}, f_{j_k}) token pair if and only if an adjacent pair is linked; that is, if $l(e_{i_k-1}, f_{j_k-1}) \in l_1^{k-1}$ or $l(e_{i_k+1}, f_{j_k+1}) \in l_1^{k-1}$. Unlike h, the features used in our implementation of a complete $P(A|E,F)$ aligner will be based on feature templates (see Sections 2.4.3 and 4.2.1). Please note that, throughout this thesis, *null* links will be considered exceptions, as the *null* location at position 0 is just an arbitrary decision for notational convenience. Therefore, it is assumed that *null* links will not activate features like the one described here unless otherwise specified.

Link sequences will be sorted in descending order according to unmodified link probability. This requires the parameter estimation process to get link and co-occurrence counts first, before it begins to collect feature counts. No efficiency is lost because, for all practical purposes, we are concerned only with counting those features that occur in the context of

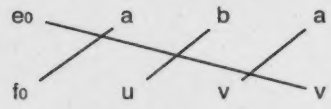


Figure 3.1: An Example Aligned Corpus

Table 3.1: Example Probability Tables

(a) Link Counts and Probabilities

$\overline{e}_{i_k}$	$\overline{f}_{j_k}$	$\lvert\overline{l}_k\rvert$	$\lvert\overline{e}_{i_k}, \overline{f}_{j_k}\rvert$	$P(l_k \lvert \overline{e}_{i_k}, \overline{f}_{j_k})$
b	u	1	1	1
a	f_0	1	2	$\frac{1}{2}$
e_0	v	1	2	$\frac{1}{2}$
a	v	1	4	$\frac{1}{4}$

(b) Feature Counts

$\overline{e}_{i_k}$	$\overline{f}_{j_k}$	$\lvert h, \overline{l}_k\rvert$	$\lvert h, \overline{e}_{i_k}, \overline{f}_{j_k}\rvert$
a	v	1	1

(c) Feature Probabilities

$\overline{e}_{i_k}$	$\overline{f}_{j_k}$	$P(h \lvert \overline{l}_k)$	$P(h \lvert \overline{e}_{i_k}, \overline{f}_{j_k})$
a	v	1	$\frac{1}{4}$

word pairs that are linked at least once in the training corpus. Therefore, this initial link counting phase would be necessary regardless of our selected ordering.

The one sentence corpus shown in Figure 3.1 would produce the probability tables shown in Table 3.1. Table 3.1(a) is created first. Co-occurrence counts are established using Equation 2.9; for example, the types a and v both occur two times in their respective sentences, therefore they co-occur $2 \times 2 = 4$ times in the sentence pair. Links can be counted in a straight-forward fashion. Link probabilities are then calculated according to Equation 3.6. These link probabilities determine the ordering of the link sequence: $\{l(b_2, u_1), l(a_1, f_0), l(e_0, v_3), l(a_3, v_2)\}$ in the training alignment[1]. Next, Tables 3.1(b) and 3.1(c) are created. Features are counted and probabilities are calculated as described in Section 3.2. The links (a_3, v_2) and (b_2, u_1) are the only links that match the feature h described above: they are mutual neighbors. However, h is active for the (a_3, v_2) token pair, while h is not active for the (b_2, u_1) token pair. This is due to our selected ordering; when $l(b_2, u_1)$

[1]We choose to break ties according to French token position

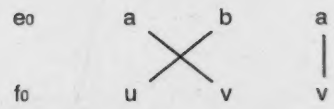


Figure 3.2: An Alternative Alignment: A'

is placed, the link $l(a_3, v_2)$ does not yet exist, according to the link sequence.

Tables 3.1(a-c) and Equation 3.5 allow one to calculate the probability of the alignment in Figure 3.1 as:

$$\begin{aligned} P(A|E,F) &= P(l(b,u)|b,u) \cdot P(l(a,f_0)|a,f_0) \cdot P(l(e_0,v)|e_0,v) \\ &\quad \cdot P(l(a,v)|a,v) \frac{P(h|l(a,v))}{P(h|a,v)} \\ &= 1 \cdot \frac{1}{2} \cdot \frac{1}{2} \cdot \frac{1}{4} \cdot \frac{1}{\frac{1}{4}} \\ &= \frac{1}{4} \end{aligned}$$

Note that, due to the simplifications made during the model derivation, the model is unable to memorize its one sentence training set. Therefore, it assigns its own training set a probability of $\frac{1}{4}$, leaving plenty of mass for other alignments. This will allow for generalization, which is necessary when dealing with previously unseen sentence pairs.

For an example of an unseen alignment, take the alignment A' shown in Figure 3.2. Using the parameter set from Table 3.1, this alternative alignment of the same sentence pair is assigned a probability of:

$$\begin{aligned} P(A|E,F) &= P(l(b,u)|b,u) \cdot P(l(a,v)|a,v) \cdot P(l(a,v)|a,v) \\ &= 1 \cdot \frac{1}{4} \cdot \frac{1}{4} \\ &= \frac{1}{16} \end{aligned}$$

Figure 3.2 does not trigger any features, even though all of its links are in close proximity of one another, as none of the links are neighbors according to our strict definition of h.

This evaluation of an unseen example brings out two interesting points. First of all, the algorithm has memorized the parts of our training set that it was allowed to access through features. It assigns low probability to all (a, v) links except for those with an established neighboring link, which are essentially free: because $\frac{1}{4} \times 1/\frac{1}{4} = 1$ they have no negative effect on $P(A|E,F)$. Similarly, all links involving (b, u) are also free. The model has reached these definite preferences after having seen only one aligned sentence

pair, indicating a strong possibility of overfitting in practical applications. We will address this issue through sampling and smoothing, both of which will be described in the following chapter. Also, one will note that the $P(A|E,F)$ product for Figure 3.1 has four terms, while the one for Figure 3.2 has only three. As any probability term has a maximum value of 1, extra terms in the equation will serve only to decrease $P(A|E,F)$. This creates a deliberate bias toward smaller alignments. When used in a complete alignment algorithm, we will count on the model penalizing unnecessary *null* links. As one can see from this example, though, if the model has enough evidence that a *null* link is appropriate, it can overcome this bias in order to prefer two links to *null* over one link between two poorly matched tokens.

Chapter 4

Alignment Algorithm

A method for evaluating the probabilities of alignments is of little use on its own. In order to create word alignments, the model must be incorporated into an alignment algorithm. This chapter will present such an algorithm designed specifically to use the model described in Chapter 3. This chapter is separate from Chapter 3 in order to emphasize that the model is not tied to a specific algorithm. Another instantiation of the algorithmic aspects described here, altering factors like the search method, constraints, or features would generate another algorithm, that may be more or less effective in aligning parallel texts. In this way, it is hoped that other researchers can take advantage of the flexible framework presented by this model to test ideas about alignment search without having to develop a new metric with which to score alignments.

This chapter will completely describe an alignment algorithm built around this model. It will begin with a high-level description of the algorithm. Subsequent sections will detail individual aspects of the algorithm, such as the features and smoothing methods used to evaluate $P(A|E,F)$ states during search, the method and constraints used to search alignment space, and the training process used to assign parameters to the $P(A|E,F)$ model.

4.1 High-level Description

The alignment algorithm presented here, dubbed ProAlign [20], takes an iterative approach to alignment creation, similar to that used by Melamed in [31] (see Section 2.3). Like competitive linking, this new algorithm does not find optimal alignments and does not do true EM parameter estimation. Unlike competitive linking, ProAlign uses a different central probability model, and makes more careful approximations when training and aligning. Many of these modifications were inspired by the approximations used by Candide Model 3 (see Section 2.2.5) and by syntactic alignment algorithms (see Section 2.5).

Given a corpus to be aligned, we first parse one half of the corpus. In our implementation, we parse the English half of the corpus using Minipar[1]. Then the parallel corpus and the English dependency trees are passed into the following high-level structure:

- Create an initial alignment of the corpus.
- Do:
  - Get co-occurrence, link and feature counts from the aligned corpus.
  - Use counts to update the probability model's parameters.
  - For each sentence pair, conduct a constrained search through alignment space for an alignment that maximizes $P(A|E,F)$. Re-align the corpus according to these new alignments.
- Repeat until tests on a validation set indicate overfitting.

In this process, every iteration before the last will be referred to as training. The final iteration will be referred to as the alignment iteration. Dependency trees are given for half of the corpus (generally assumed to be the English half) so that the model can use linguistic aspects of sentences as features, and so the alignment search can be constrained using linguistic constraints.

The details of the components of this algorithm are described in the order in which they are most easily understood. The process which creates the initial alignment is described in Section 4.4.1. In order to count links and features, the algorithm samples from a set of possible alignments as described in Section 4.4.2. Transforming counts into parameters has been described completely in Section 3.2, though these parameters will also be smoothed (Section 4.2.2). The alignment search consists of a method for traversing alignment space (Section 4.3) and a method for evaluating individual alignments (Chapter 3 and Section 4.2).

4.2 Probability Evaluation

The model presented in Chapter 3 is fairly general. This section will detail the features selected to be used in this instantiation of the $P(A|E,F)$ model. It will also discuss two smoothing methods used in order to make the model more effective in practice. The model, with the described features and smoothing method, will be used to evaluate states during the alignment search.

[1] available at http://www.cs.ualberta.ca/~lindek/minipar.htm

4.2.1 Features

This section describes the two feature templates that were used in the ProAlign word alignment algorithm. The development of these features involved experiments with several types of feature templates. The templates that were found to be most effective were those that adjusted link probability based on other links established in the alignment. Features produced by the first feature template account for the presence of surrounding links. These are called **adjacency features**. The second template tracks dependency links present in the parsed sentence, creating **dependency features**. Each will be described in turn.

The adjacency feature template creates features that will alter a potential link's context ratio according to those links that exist around it. As in HMM-based alignment, this template takes advantage of the observation that words close to each other in the source language tend to remain close to each other when translated [41]. Therefore, a potential link might be considered more likely if it has several neighboring links. Since the feature model presented in Chapter 3 is quite inexpensive to train (it requires only counting features as they occur with both word pairs and links), it is capable of tracking a very large number of features. To take advantage of this, the template will be made so that each pair of relative positions triggers a different feature. The features can be made even more specific by tracking the English word involved in the neighboring link. To capture all of this information, the feature template takes the following form: for any word pair (e_i, f_j) with a context C_k, if another link $l(e_{i'}, f_{j'})$ exists within a two word window of the pair (that is, if $i-2 \leq i' \leq i+2$ and $j-2 \leq j' \leq j+2$), then we say that the feature $h_a[i-i', j-j', \bar{e}_{i'}]$ is active for this context. This is denoted as $h_a[i-i', j-j', \bar{e}_{i'}](C_k) = 1$. We have adopted a square bracket notation for these feature functions only because subscripts are unwieldy with such a large number of variables.

The second feature template creates dependency features. These features use the English parse tree to capture regularities among grammatical relations between languages. For example, when dealing with French and English, the location of a determiner with respect to its governor[2] is never swapped during translation: in both languages, determiners come before the words they modify. Conversely, the location of an adjective with respect to its governor is swapped frequently; adjectives tend to come before their noun in English, and after their noun in French. In order to allow the model to learn these types of preferences, the template for dependency features is designed to capture information about

[2]The parent node in the dependency tree.

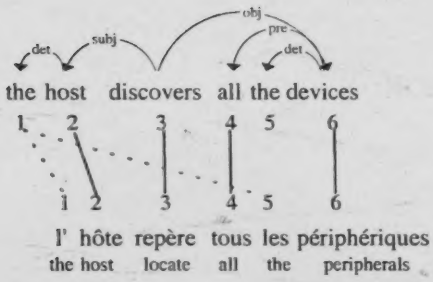


Figure 4.1: Feature Extraction Example

relative French positions and relationship types. English positions are not needed because in this case the dependency data actually gives more information than relative word position. For any word pair (e_i, f_j), let $e_{i'}$ be the governor of e_i according to our dependency tree. Let *rel* be the label of the relationship between child and parent. If a link $l(e_{i'}, f_{j'})$ exists, then we say that the feature $h_d[j - j', rel]$ is active for this context.

Take for example Figure 4.1 which shows a partial alignment with all links completed except for those involving the English type *the*. Given this sentence pair and English parse tree, one can extract features of both types to assist in the alignment of the_1. The token pair (the_1, l') will have an active adjacency feature $h_a[+1, +1, host]$ as well as a dependency feature $h_d[-1, det]$. Because both of these features will tend to occur in the context of the type pair (the, l') only when the words are linked, in a trained model these two features will work together to increase the probability of making a (correct) link between this word pair. In contrast, the incorrect link (the_1, les) will have only one active feature, $h_d[+3, det]$. Most determiners are located before their governors, so this feature will not occur frequently in the presence of linked instances of the type pair (the, les). Therefore, the presence of this feature will lower the link probability.

4.2.2 Smoothing

As was discussed in the example in Chapter 3, the $P(A|E, F)$ model used by ProAlign can overfit in the presence of sparse data. To deal with this problem, two smoothing methods are used: one for link probabilities, and another to smooth context ratios directly. This section will describe both methods in turn.

Link Probability Smoothing

The link probability term, denoted by $P(l_k|\overline{e}_{i_k}, \overline{f}_{j_k})$ assigns a base probability to linking a token pair according to the types that are in the pair. Unfortunately, some type pairs are seen only infrequently, and those pairs may be accidentally linked from time to time. Without smoothing, this can cause large inaccuracies in model parameters. Take the English-French type pair (*black*, *noir*), for example. Its two types are words that are considered to be good translations of one another; therefore, one would hope that this pair would be assigned a high link probability. In the development corpus that was used when designing ProAlign, this pair of types co-occurred 58 times and were linked 52 times, setting the unsmoothed link probability to $P(l_k|black, noir) = \frac{52}{58} = 0.9$. In contrast, the word types *black* and *gras* (which means in English bold or fat) co-occurred twice and were linked twice, giving it a link probability of $P(l_k|black, gras) = \frac{2}{2} = 1$. Intuitively, $P(l_k|black, noir)$ should be much higher than $P(l_k|black, gras)$, as the former has much more evidence supporting it. This problem can be addressed with a simple m-estimate smoothing method [34]. With this method, the smoothed link score is calculated as:

$$P(l_k|\overline{e}_{i_k}, \overline{f}_{j_k}) = \frac{links(e, f) + mp}{cooc(e, f) + m} \tag{4.1}$$

In ProAlign, the values m=5 and p=0.05 are used. With smoothing, the link probability of (*black*, *noir*) is lowered to 0.83, while the link probability of (*black*, *gras*) is lowered much further to 0.3.

Context Ratio Smoothing

The context ratio is calculated as a product of feature ratios that modify link probability according to the features found in the context of a word pair. Each feature ratio has the form $\frac{P(h|l_k)}{P(h|\overline{e}_{i_k}, \overline{f}_{j_k})}$. Each of the probabilities involved in the ratio can be smoothed in a straightforward manner using m-estimate smoothing, as was done above for link probabilities. This solves some problems with overfitting, but problems can still occur when a feature occurs very infrequently in the context of a given word pair. For example, take the antagonistic case where a feature h only occurs in the context of a word pair once, and in that one instance, the word pair is (incorrectly) linked. Furthermore, suppose that the two types co-occur three hundred times throughout the corpus, and they are linked only twice throughout. The resulting feature ratios produced by this example with several alternative smoothing methods are shown in Table 4.1. The completely unsmoothed feature ratio would produce an unreasonably large ratio, boosting link probability by a factor of 150 whenever h is

Smoothing?	$P(h\|\overline{l}_k)$	$P(h\|\overline{e}_{i_k}, \overline{f}_{j_k})$	Ratio
None	$\frac{1}{2} = 0.5$	$\frac{1}{300} = 0.003$	$\frac{0.5}{0.003} = 150$
Probs only	$\frac{1+0.01}{3} = 0.337$	$\frac{1+0.01}{301} = 0.003$	$\frac{0.337}{0.003} = 100$
Probs & Ratio	0.337	0.003	$1 + \left(\frac{1}{1+1} \cdot 99\right) = 50.5$

Table 4.1: Context Ratio Smoothing Example

present in the context of the type pair. Aggressively smoothing the individual probabilities (with $m = 1$ and $p = 0.01$) produces some improvement, as can be seen in the *Probs only* row of the table, but the ratio is still very large. We can further reduce this ratio by smoothing the ratio directly, according to the number of times we have seen the feature in the context of this pair.

The goal of ratio smoothing is to pull the ratio closer to 1 when evidence for a large ratio is lacking. This can be achieved using a direct, but theoretically unmotivated method which does exactly that. Let $c_h = |h, \overline{e}_{i_k}, \overline{f}_{j_k}|$ be the number of times a feature h occurs in the context of the word pair in question, and let r be the ratio produced by dividing the smoothed feature probabilities, ie: the ratio from the *Prob only* row in Table 4.1. The ratio smoothing method used by ProAlign pushes the feature ratio toward 1 according to a factor of $c_h/(c_h + 1)$ with the following smoothing formula:

$$newRatio = \begin{cases} 1 + (r-1)\left(\frac{c_h}{c_h+1}\right) & \text{if } r > 1 \\ \frac{1}{1+(\frac{1}{r}-1)\left(\frac{c_h}{c_h+1}\right)} & \text{if } r < 1 \\ r & \text{if } r = 1 \end{cases} \tag{4.2}$$

Thus, features which have been seen many times in the context of the word pair are changed only slightly by this method, because $c_h/(c_h + 1)$ approaches 1 quickly as c_h grows. However, this method does greatly reduce those features that have only a few occurrences supporting them. To complete our example, one can see in Table 4.1 that the ratio-smoothed ratio is reduced to 50.5. This is still high enough to cause overfitting, but this example is an artificially antagonistic case, and the ratio has been greatly reduced from its original unsmoothed value. Informal development tests have shown ratio smoothing to increase performance in practice.

4.3 Search

Given the $P(A|E, F)$ model and the features and smoothing methods described above, ProAlign is now equipped with a method to evaluate any number of alignments between

two sentences. In order to achieve the objective of building high quality alignments, the challenge remains to find an alignment A that maximizes $P(A|E,F)$. To achieve this goal, ProAlign conducts a search through alignment space. As was discussed in the initial description of Candide Model 1 in Section 2.2.2, the number of possible alignments for a pair of sentences is too large to work with explicitly using current technology. So, like Candide Model 3, ProAlign explores alignment space heuristically. To do so, the search algorithm employed is a beam search with constant beam and agenda size. To make alignment space more manageable, a number of constraints are employed to limit the search. These constraints will be discussed in Section 4.3.1. Each state in alignment space represents a partial alignment. A transition is defined as the addition of a single link to the current state. A transition is valid if the addition results in a (partial) alignment that does not violate any constraints. The start state is the empty alignment, where all words in E and F are implicitly linked to null. A terminal state is a state in which no more links can be added without violating a constraint or reducing $P(A|E,F)$. The goal of the best-first search is to find the terminal state with the highest probability.

For the purpose of guiding ProAlign's best-first search, non-terminal states in alignment space are evaluated according to a greedy completion of the partial alignment. ProAlign builds this completion by adding valid links in the order of their unmodified link probabilities $P(l_k|\overline{e}_{i_k},\overline{f}_{j_k})$ until no more links can be added. The score the state receives is the probability of its greedy completion. These completions are saved for later use in alignment sampling (see Section 4.4.2).

4.3.1 Constraints

The reader may have noted that ProAlign's $P(A|E,F)$ alignment model has very few factors to explicitly prevent undesirable alignments, such as having all French words align to the same English word. This is in contrast to Candide Models 2 and above, all of which track relatively complex statistics to learn about factors relevant to alignment such as token position and fertility. ProAlign, like Candide Model 1 and Melamed's models of translational equivalence, is concerned primarily with measuring the strength of translation probability between two word types. Like Melamed's models, ProAlign relies on constraints to guide the alignment search to alignments which are linguistically reasonable. The two constraints used by ProAlign, the one-to-one constraint and the cohesion constraint, are described below.

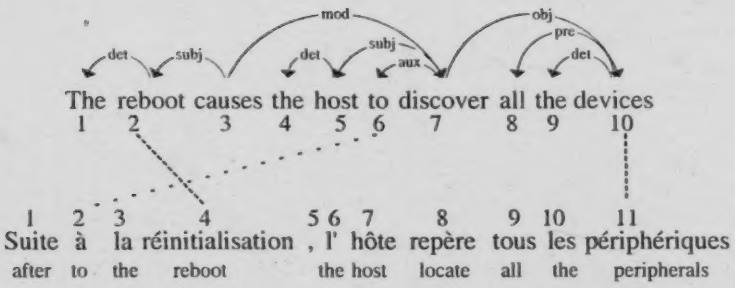

mod
obj
pre
det
subj
det
subj
aux
det
The reboot causes the host to discover all the devices
1 2 3 4 5 6 7 8 9 10
1 2 3 4 5 6 7 8 9 10 11
Suite à la réinitialisation , l' hôte repère tous les périphériques
after to the reboot the host locate all the peripherals

Figure 4.2: A Cohesion Constraint Violation

One-to-one

ProAlign makes use of the one-to-one constraint as it is described in [31]. In both sentences, every token (except the *null* words e_0 and f_0) participates in exactly one link. That is, each token is linked either to one other token or to *null*. The constraint removes the need to track certain aspects of alignments such as fertility, but it maintains the algorithm's ability to use the pigeon hole principle to infer placement of future links based on previously established links. The one-to-one constraint is not held universally in word alignments created by human translators. However, in the case of English-French sentence pairs, it is held often enough so that enforcing it does not have a catastrophic effect on ProAlign's ability to recall true links.

Cohesion

The notion of using syntax as an alignment guide was sketched briefly in Section 2.5. There we presented the hypothesis that phrasal cohesion is conserved during translation [12]. ProAlign's second constraint, called the cohesion constraint [21], forces all alignments to maintain phrasal cohesion. If two phrases are disjoint in the English sentence, ProAlign will not be allowed to map them to overlapping intervals in the French sentence. For example, in Figure 4.2, the cohesion constraint will rule out the possibility of aligning *to* with *à*. The phrases *the reboot* and *the host to discover all the devices* are disjoint, but the partial alignment in Figure 4.2 maps them to overlapping intervals. It was shown in [21] that enforcing this constraint can increase alignment accuracy significantly. Checking that phrasal cohesion is conserved is not trivial; the remainder of this subsection will specify the cohesion constraint mathematically, and describe the cohesion checking algorithm used by ProAlign.

In order to define cohesion precisely, one must first define the notion of induced phrasal

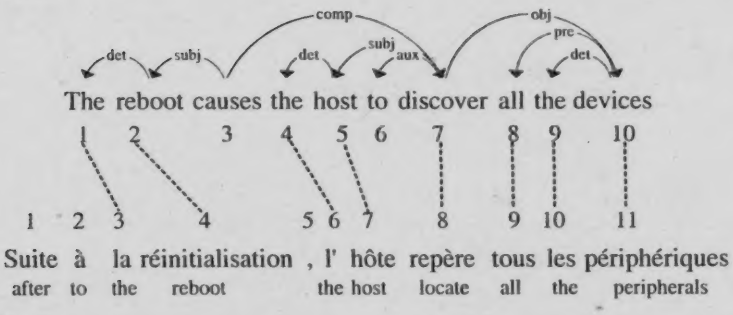

comp
obj
pre
det
subj
det
subj
aux
det
The reboot causes the host to discover all the devices
1 2 3 4 5 6 7 8 9 10
1 2 3 4 5 6 7 8 9 10 11
Suite à la réinitialisation , l' hôte repère tous les périphériques
after to the reboot the host locate all the peripherals

Figure 4.3: An Aligned Sentence Pair with English Dependency Tree

spans [12]. Once phrasal spans have been defined, one can then check for overlapping spans, which represent cohesion violations. Let T_E be the dependency tree of the English sentence. Let $T_E(e_i)$ be the subtree of T_E rooted at e_i. The **phrase span** of e_i, $spanP(e_i, T_E, A)$, is the image of the English phrase headed by e_i in F given a (partial) alignment A. More precisely, $spanP(e_i, T_E, A) = [k_1, k_2]$, where

$$k_1 = \min\{j | l(e_u, f_j) \in A, e_u \in T_E(e_i)\}$$
$$k_2 = \max\{j | l(e_u, f_j) \in A, e_u \in T_E(e_i)\}$$

The **head span** is the image of e_i itself. We define $spanH(e_i, T_E, A) = [k_1, k_2]$, where

$$k_1 = \min\{j | l(e_i, f_j) \in A\}$$
$$k_2 = \max\{j | l(e_i, f_j) \in A\}$$

In Figure 4.3, the phrase span of the node *discover* is [6, 11] and the head span is [8, 8]; the phrase span of the node *reboot* is [3, 4] and the head span is [4, 4]. The word *cause* has a phrase span of [3,11] and its head span is the empty set $\emptyset$.

With these definitions of phrase and head spans, we define two notions of overlap, originally introduced in [12] as **crossings**. Given a head node e_h and its modifier e_m, a **head-modifier overlap** occurs when:

$$spanH(e_h, T_E, A) \cap spanP(e_m, T_E, A) \neq \emptyset$$

Given two nodes e_{m_1} and e_{m_2} which both modify the same head node, a **modifier-modifier overlap** occurs when:

$$spanP(e_{m_1}, T_E, A) \cap spanP(e_{m_2}, T_E, A) \neq \emptyset$$

We say an alignment is cohesive with respect to T_E if it does not introduce any head-modifier or modifier-modifier overlaps. Returning to our example, the alignment A in

Figure 4.2 is not cohesive. This is because there is a modifier-modifier overlap between $spanP(reboot, T_E, A) = [4, 4]$ and $spanP(discover, T_E, A)$=[2, 11]. If an alignment A' violates the cohesion constraint, any alignment A that is a superset of A' will also violate the cohesion constraint. This is because any pair of nodes that have overlapping spans in A' will still have overlapping spans in A.

With these definitions of spans and overlaps in place, it becomes relatively simple to design an algorithm that checks whether adding an individual link $l(e_i, f_j)$ to a partial alignment A causes a cohesion constraint violation. Let $e_{p_0}, e_{p_1}, e_{p_2}, \ldots$ be a sequence of nodes in T_E such that e_{p_0}=e_i and e_{p_k}=$parentOf(e_{p_{k-1}})$ $(k = 1, 2, \ldots)$. The algorithm to check the cohesion constraint after adding $l(e_i, f_j)$ to A is as follows:

1. For all $k \geq 0$, update the *spanP* and the *spanH* of e_{p_k} to include j.
2. For each e_{p_k} $(k > 0)$, check for a modifier-modifier overlap between the updated phrase span of $e_{p_{k-1}}$ and the phrase span of each of the other children of e_{p_k}.
3. For each e_{p_k} $(k > 0)$, check for a head-modifier overlap between the updated phrase span of $e_{p_{k-1}}$ and the head span of e_{p_k}.
4. If an overlap is found, return true (the constraint is violated). Otherwise, return false.

Since adding a link to a partial alignment corresponds to a state transition in ProAlign's best-first search, this algorithm will serve as a method to check whether or not a given transition is valid according to the cohesion constraint.

4.4 Training

As was stated in Section 3.2, ProAlign's probability model needs an initial alignment in order to create its probability tables. Since this alignment will be noisy (the objective is to iteratively improve alignments), in order to avoid having the model relearn mistakes, it helps to sample from a set of possible alignments for each sentence, rather than train from one very likely alignment. The following sections describe the method ProAlign uses to create its initial alignment, as well as the sampling method used in training.

4.4.1 Initial Alignment

ProAlign produces an initial alignment using the same constrained alignment search described in Section 4.3, with an alternative state scoring metric. Rather than searching for

the alignment that maximizes $P(A|E,F)$, ProAlign's initial alignment method attempts to maximize the sum of the ϕ^2 correlation scores (see Section 2.3) of the word pairs linked by the alignment. Otherwise, the search remains identical to the one described above. This search and scoring method produce reasonable one-to-one word alignments that ProAlign can refine using the $P(A|E,F)$ probability model.

4.4.2 Sampling

There is currently no method we are aware of to implicitly enumerate all of the possible alignments for a sentence pair, where every alignment enumerated must obey ProAlign's two constraints. These constraints tie words in such a way that the space of alignments cannot be enumerated easily as in Candide Models 1 and 2. It may be possible to do such an implicit enumeration using synchronous parsing, but we have yet to explore that option (see Section 6.3). Taking our lead from Candide Models 3, 4 and 5, ProAlign will sample from the space of those high-probability alignments that do not violate its constraints. It will then redistribute the $P(A|E,F)$ model's probability mass evenly among its sample.

ProAlign's best-first search evaluates each non-terminal state in alignment space according to a heuristic completion of the state's partial alignment (see Section 4.3). For a given sentence pair, ProAlign's sample S of possible alignments will be the most probable alignment generated by the search, plus the greedy completions of all of the states visited during search. In this way, it samples from a large set of complete alignments which do not violate any of its constraints. This set can potentially contain duplicates, as many partial alignments can have the same heuristic completion. As in the Candide models, any sampling method that concentrates on complete, valid and high probability alignments will accomplish the same task.

When collecting the statistics needed to calculate $P(A|E,F)$ from ProAlign's initial ϕ^2 alignment, each $s \in S$ is given a uniform weight. This is reasonable, as there are no probability estimates available at this point: each alignment is graded only by a sum of correlation scores. However, duplication in the sample can still allow some, more frequently selected completions to be given an effectively larger weight. When training from the alignments produced by the later ProAlign iterations which use the $P(A|E,F)$ model, ProAlign weights alignments in the sample according to $P(s|E,F)$, normalized so that $\sum_{s \in S} P(s|E,F) = 1$. Once alignments have been weighted, the parameter estimation phase can then simply count the links and features in an alignment $s \in S$ according to s's alignment weight.

Chapter 5

Experimental Design and Results

The goal of the ProAlign system and its $P(A|E,F)$ probability model is to create high quality word alignments from parallel text. This chapter will describe a series of experiments designed to evaluate ProAlign's success in achieving this goal by evaluating the alignments that it produces. All experiments will make use of a common evaluation framework which will be described in the following section. The subsequent sections will detail the form and results of experiments designed to test system performance, algorithmic contributions, and model contributions respectively. These sections will also provide analysis of the results they present.

5.1 Evaluation Procedure

An advantage of working in an established and competitive research area is that evaluation frameworks, data sets and tools have often already been established in the community. Although word alignment is a relatively young subfield of machine translation, it has grown quickly and has a fair amount of established infrastructure. We took advantage of this infrastructure in order to ensure that our results were comparable with the rest of the community.

The current standard for evaluating word alignments was first proposed in [37]. Under this system, an alignment is seen as a set of links between positions. This set of links is then compared to a gold standard created by humans familiar with both languages. The humans are instructed to manually annotate a sentence pair with links, leaving words linked to *null* unmarked. Since even human translators can be uncertain of the "correct" way to align some sentence pairs, a system to capture uncertainty and multiple possibilities was developed. The annotator labels each link he or she adds to the sentence as either **sure** or **possible**. Sure links occur when an annotator is certain that a link is both correct and

necessary. Possible links occur when a translator feels that there are multiple possible interpretations, or that a link may or may not exist in a correct alignment. A testing set is annotated by multiple translators, and the annotations are merged into one answer key. The set of sure links in the answer key is taken to be only those sure links which all annotators agree upon. The final set of possible links is the union of all sure or possible links drawn by any annotator. Therefore, all sure links are also possible.

Given an machine-generated alignment of sentences for which gold standard alignments exist, one can calculate a number of performance metrics, also defined in [37]. These measures include precision and recall, which are familiar to the information retrieval and machine learning communities, and alignment error rate (AER), which is specific to this word alignment evaluation procedure. Let A be the set of links in the alignment to be evaluated, and let P and S be the set of possible and sure links present in the gold standard alignment. Recall, the ability of the algorithm to generate those links deemed necessary by the annotators, can be measured as:

$$\text{recall} = \frac{|A \cap S|}{|S|}$$

Precision, the ability of the algorithm to only generate links which are considered possible by the annotators, can be measured as:

$$\text{precision} = \frac{|A \cap P|}{|A|}$$

Finally, alignment error rate, which is a measure of the number of errors present in an alignment, can be measured as:

$$\text{AER} = 1 - \frac{|A \cap S| + |A \cap P|}{|S| + |A|}$$

During system development, ProAlign was trained with a French-English corpus with 200K sentences, provided by Sun Microsystems. The particulars of the algorithm were tuned according to ProAlign's performance on a 100 sentence answer key drawn from the same source and excluded from the training set. This key was annotated in house without distinguishing between sure and possible links. Therefore, all links were considered both sure and possible. All in-house experiments were conducted after training ProAlign with a French-English data set of 50K sentence pairs drawn from the sentence-aligned Canadian Hansards. ProAlign was then evaluated against 500 manually aligned sentences also drawn from the Canadian Hansards, but excluded from the training set. The answer key was annotated externally by a pair of annotators using both sure and possible alignments. The training and testing sets for the in-house experiments were kindly provided by Franz Och.

5.2 Comparisons to the State of the Art

Two experiments were conducted in order to determine whether or not ProAlign improves upon the current state of the art in word alignment. The first experiment, conducted in house, compares ProAlign alignments to those produced by GIZA++ when training on a relatively small corpus. The second experiment summarizes ProAlign's performance in a word alignment competition conducted as a part of a recent workshop on parallel text, which involved a much larger corpus, as well as a language pair other than French-English.

5.2.1 In-house Experiments

We used the experimental data set described in Section 5.1 to evaluate ProAlign's alignment quality. Results on the same data set using the state of the art translation model, GIZA++ have been previously reported in [37]. GIZA++ is an implementation of Candide Model 4, which employs an HMM in place of Candide Model 2 in the Candide model hierarchy. Table 5.1 compares ProAlign's scores with the results in [37]. The rows *Candide-4 F→E* and *Candide-4 E→F* are the results obtained by Candide Model 4 when treating French as the source and English as the target or vice versa. The row *Candide-4 Intersect* shows the results obtained by taking the intersection of the alignments produced by *Candide-4 E→F* and *Candide-4 F→E*. The row *Candide-4 Refined* shows results obtained by refining the intersection of alignments in order to increase recall. GIZA++, alignment intersection, and refined intersection are described in greater detail in Section 2.2.7.

ProAlign achieved over 44% relative error reduction when compared with Candide-4 used in either direction and a 25% relative error rate reduction when compared with *Candide-4 Refined*. It also achieved a slight relative error reduction when compared with *Candide-4 Intersect*. This demonstrates that ProAlign is competitive with the methods described in [37]. Furthermore, the results produced by ProAlign's single alignment model are competitive with the combination of two Candide models. For large corpora, the difference between training two models and training one can mean extra hours, or even days, of training time.

In Table 5.1, one can see that ProAlign is high precision, low recall. This was expected as ProAlign uses the one-to-one constraint, which rules out many of the possible alignments present in the evaluation data. This systematically reduces recall. The fact that precision is so high is very encouraging. It indicates that ProAlign's model and constraints are working together to select only those links for which the model is certain. It also indicates that these

Table 5.1: Comparison with GIZA++ Candide

Method	Prec (%)	Rec (%)	AER (%)
ProAlign	**95.7**	86.4	**8.7**
Candide-4 Intersect	95.7	85.6	9.0
Candide-4 Refined	85.9	**92.3**	11.7
Candide-4 F→E	80.5	91.2	15.6
Candide-4 E→F	80.0	90.8	16.0

links would have very little noise, if they were used as input for an appropriate translation algorithm or some sort of supervised learning algorithm. The closest competing algorithm, *Candide-4 Intersect* shares a few factors in common with ProAlign. They are both symmetric (intersecting two Candide models removes the asymmetry inherent in Candide) and they are both one-to-one. This suggests that these qualities are favorable when doing alignment, at least with this language pair and data set.

5.2.2 Word Alignment Shared Task

At the meeting of the North American chapter of the Association for Computational Linguistics in 2003, a workshop entitled, "Building and Using Parallel Texts: Data Driven Machine Translation and Beyond," held a shared task in word alignment. We entered a modified version of ProAlign into this shared task, the results of which are reported in [33] and are summarized below. The shared task compared the results of seven different teams from around the world in two subtasks in word alignment. One subtask measured performance on a large corpus of a familiar language pair, French-English. This training corpus consisted of 1 million sentence pairs drawn from the Canadian Hansards. The testing set was roughly 450 sentences, also drawn from the Hansards. The second subtask measured performance with a smaller training set, on a less familiar language pair, Romanian-English. This training corpus consisted of roughly 50 thousand sentences, drawn from a diverse set of sources including newspaper text, the Romanian constitution and a translation of George Orwell's "1984". The testing set consisted of 250 manually aligned sentences. In the Romanian-English answer key, no distinction was made between sure and possible alignments. Only five teams competed in the French-English task, while all seven teams competed in the Romanian-English task.

The version of ProAlign used in this task was slightly modified from the one described in Chapter 4. A different initial alignment algorithm was employed. One of the strengths

Table 5.2: Shared Task Results: French-English

Method	Institution	Prec (%)	Rec (%)	AER (%)
ProAlign	(University of Alberta)	**96.49**	91.48	**5.71**
XRCE	(Xerox Research Center Europe)	90.09	**93.81**	8.53
Ralign	(Université de Montréal)	77.56	80.61	18.50
BiBr	(LTI Carnegie Melon University)	66.65	82.42	28.01
UMD	(University of Minnesota, Duluth)	59.69	64.66	38.47

of the $P(A|E,F)$ model presented in this thesis is that it is designed to improve an existing word alignment. The alignment improved by ProAlign in our controlled experiments is based on the well-understood ϕ^2 correlation metric, to make the results more understandable to the community. However, our team also has access to a more powerful initial alignment generator, described in [19], which uses linguistic heuristics to guide an alignment search. By starting with alignments generated by this more complex system, and then iteratively improving using ProAlign, we were able to make ProAlign more competitive in this task. To further improve results, we employed a heuristic alignment expansion on ProAlign's results in order to capture correct links that may violate ProAlign's search constraints. This expansion is also explained in [19].

A summary of each team's best result (multiple system submissions were allowed) in the French-English task is shown Table 5.2. One can see that ProAlign is the most precise algorithm by a large margin, and that it has the lowest error rate. Its closest competitor was submitted by the Xerox Research Center Europe (XRCE). The method employed by XRCE was simply to use GIZA++ without modification [9], and without using alignment refinement or intersection. They chose to train GIZA++ using only half the corpus, presumably due to time constraints (they report results with the entire training corpus in [9]). ProAlign improves upon XRCE's alignment error rate with a relative reduction of 33%. It is encouraging to see that ProAlign is able to continue to out-perform GIZA++, even when both methods are trained using a substantially larger corpus than the one used in our in house experiments. GIZA++ is known to use large corpora effectively [37].

A summary of each team's best result in the Romanian-English task is shown in Table 5.3. Once again, ProAlign is the most precise, but it has been surpassed in alignment error rate by XRCE. As before, XRCE's entry is an unmodified version of GIZA++, without intersection or refinement. However, in this task the two top AER scores are very close. XRCE's AER score represents only a 1.7% relative improvement in alignment error rate

Table 5.3: Shared Task Results: Romanian-English

Method	Institution	Prec (%)	Rec (%)	AER (%)
XRCE	(Xerox Research Center Europe)	82.65	**62.44**	**28.86**
ProAlign	(University of Alberta)	**88.22**	58.91	29.36
RACAI	(Romanian Academy Institute)	81.29	60.26	30.79
Ralign	(Université de Montréal)	63.63	45.06	35.24
BiBr	(LTI Carnegie Melon University)	59.60	55.75	41.39
UMD	(University of Minnesota, Duluth)	58.29	49.99	46.18
Fourday	(MITRE Corporation)	52.83	42.86	52.67

over ProAlign's results. Since ProAlign relies on the cohesion constraint to guide link placement and XRCE's GIZA++ system does not, it is encouraging to see that ProAlign remains competeive with GIZA++, even on a completely different language pair.

There are several factors to note regarding the Romanian-English task. First of all, the scores for all teams are substantially less encouraging than those received in the previous French-English task. The smaller training set is cited in [33] as the explanation for the decrease in quality. However, the training set in this task is roughly the same size as our in house experimental data set, with which ProAlign achieves excellent scores. We feel that there are more likely explanations for the differences in score. One explanation could be the heterogeneous nature of the Romanian-English corpus, which was drawn from news, political and fictional documents, all of which may have very different styles of language use. This is intuitively more difficult than learning to translate parliamentary documents by examining only parliamentary documents, as is done in the case of the French-English Hansards. The higher error rates may have also occurred due to noise in the corpus; our team found the newspaper section of the corpus to be especially noisy. Another factor that may have reduced scores is differences in test-set annotation styles between the French-English and Romanian-English tasks, as the Romanian-English answer key was aligned by a different group that may have followed slightly different guidelines. Finally, the Romanian-English language pair may simply be more difficult to decipher using statistical techniques. On a related note, when comparing the methods that competed in this task, it is important to note that the test set in the Romanian-English task was annotated using only sure links. It has been our experience that when there is no distinction between sure and possible links, AER tends to favor high recall methods. If $S = P$, then the formula for AER becomes:

$$\text{AER} = 1 - \frac{2 \cdot |A \cap P|}{|P| + |A|}$$

Table 5.4: Evaluation of Components

With Cohesion Constraint			
Algorithm	Prec (%)	Rec (%)	AER (%)
initial (ϕ^2)	88.9	84.6	13.1
without features	93.7	84.8	10.5
with h_d only	95.6	85.4	9.3
with h_a only	95.9	85.8	9.0
with h_a and h_d	95.7	86.4	8.7

Without Cohesion Constraint			
Algorithm	Prec (%)	Rec (%)	AER (%)
initial (ϕ^2)	81.4	84.1	17.4
with h_a and h_d	87.4	85.3	13.6

Effectively, all of the links in the answer key become certain and necessary. This places primarily one-to-one methods such as ProAlign at a disadvantage, as ProAlign's alignment search cannot form some of the link combinations required to exactly match the annotators' alignments. Therefore, the higher recall XRCE method may have passed ProAlign in alignment error rate only because of this irregularity in the task's answer key.

5.3 Contributions of Algorithmic Components

At this point we have established that ProAlign is competitive with the current state of the art, and under the correct circumstances, it can surpass the state of the art in terms of precision and alignment error rate. However, ProAlign is a complicated search algorithm built around a model that can take into account many types of features, and it is not clear how much each algorithmic component contributes to the success of the system. Therefore, we conducted a number of experiments using the experimental data set described in Section 5.1 in order to determine how much each major component of the ProAlign algorithm contributes to alignment quality. The components included in this experiment are the cohesion constraint, the model itself, dependency features (h_d), and adjacency features (h_a).

Table 5.4 shows the contributions of algorithmic components to ProAlign's performance. The first block shows results where the cohesion constraint was enforced, while the second shows results where the constraint is relaxed. Within blocks, the *initial (ϕ^2)* row is the score for the algorithm described in Section 4.4.1 that generates ProAlign's initial

alignment. The *without features* row shows the score after training using the $P(A|E,F)$ model with an empty feature set. The rows *with h_d only* and *with h_a only* describe the scores after training using only dependency and adjacency features respectively. Finally *with h_a and h_d* shows results with the complete ProAlign feature model.

By comparing the two blocks in Table 5.4, one can see that the most dramatic contributor is the cohesion constraint. It reduces the complete model's AER from 13.6 to 8.7, for a relative reduction of 36%. In fact, the initial alignment using cohesion constraint receives a lower AER score (13.1) than the best achievable without cohesion (13.6). One can also see that the $P(A|E,F)$ model's ability to iteratively improve word alignments is not dependent on the cohesion constraint. Even without the constraint, the complete model is able to reduce the initial alignment scoring 17.4 to 13.6 AER, for a relative reduction of 21%.

Taking the cohesion constraint as given and examining the first block alone, one can see that the $P(A|E,F)$ model in its simplest (feature-free) form is capable of producing a significant improvement in alignment quality, reducing initial AER from 13.1 to 10.5 for relative reduction of 19.8%. The two feature templates applied alone each provide significant contributions, with the adjacency features being slightly more influential. The final row, representing the complete algorithm, shows that both feature templates can work together to create a greater improvement, despite the independence assumptions made in the $P(A|E,F)$ feature model. The relative reduction in error rate from the initial alignment to an alignment created using the model and all features is 33.6%.

5.4 Validating the Probability Model

Even though we have compared ProAlign to alignments created using Candide statistical models (GIZA++ and XRCE), it is not clear if the new $P(A|E,F)$ model is essential to ProAlign's performance. It is possible that another model that measures $P(A|E,F)$ would perform just as well or better in refining an initial alignment. This experiment aims to determine if we could have achieved similar results using the same initial alignment and constrained search algorithm with an alternative model of $P(A|E,F)$.

Without using any features, our model is similar to Candide's Model 1 (Section 2.2.2). Both take into account only the word types that participate in a given link. Candide Model 1 uses $P_t(\overline{f}|\overline{e})$, the probability of $\overline{f}$ being generated by $\overline{e}$. Conversely, our model in built around $P(l_k|\overline{e}_{i_k},\overline{f}_k)$, the probability of a link existing between $\overline{e}_{i_k}$ and $\overline{f}_{j_k}$. In this experiment, we set Model 1 translation probabilities according to our initial ϕ^2 alignment,

Table 5.5: $P(A|E,F)$ Model versus Candide Model 1

Algorithm	Prec	Rec	AER
initial (ϕ^2)	88.9	84.6	13.1
Our $P(A\|E,F)$ model	93.7	84.8	10.5
Candide Model 1	89.2	83.0	13.7

sampling as we described in Section 4.4.2. We then use the Model 1 measure of alignment probability:

$$P(A|E,F) \propto \prod_{j=1}^{n} P_t(\overline{f}_j|\overline{e}_{a_j})$$

to evaluate candidate alignments in a search that is otherwise identical to the ProAlign algorithm. We ran this Model 1 alignment refinement on our experimental data set (Section 5.1) for three iterations and recorded the best results that it achieved.

It is clear from Table 5.5 that refining an initial ϕ^2 alignment using IBM's Model 1 is less effective than using our $P(A|E,F)$ model in the same manner. In fact, the Model 1 refinement receives a lower score than the initial alignment. We can therefore conclude that we were justified in creating a new probability model for this task.

Chapter 6

Future Work

In process of writing this thesis and disseminating its results to the natural language processing community, several ideas have come up to improve ProAlign's performance. This chapter will describe three ways in which the algorithm presented here could be extended. The first involves softening the one-to-one and cohesion constraints, so they can be violated when it is appropriate. The second examines replacing the current link probability model with one implemented using maximum entropy. Finally, we would like to explore the use of synchronous parsing as a search method.

6.1 Soft Constraints

ProAlign employs two hard constraints, the one-to-one constraint and the cohesion constraint. The one-to-one constraint is the most limiting, stopping the algorithm from correctly linking any term which has a multiple-token translation. The cohesion constraint, though it is held roughly 90% of the time in French-English translation [12], could also be softened to improve alignment recall in those cases where the English sentence is parsed incorrectly. This section will briefly describe how we plan to address these two issues. We are currently simulating having soft constraints by using a heuristic post-processing step, described in [19], to add links that may violate constraints to ProAlign's final alignment. This process is not very effective, and we would like to replace it with a more principled solution.

ProAlign, as it is described in this thesis is incapable of creating alignments that are not one-to-one. The $P(A|E,F)$ model it is built around, however is not limited in the same manner. The model is currently capable of creating many-to-one alignments so long as the *null* probabilities of the words added on the "many" side are less than the probabilities of the links that would be created. Under the current implementation, the training corpus is

one-to-one, which gives our model no opportunity to learn many-to-one alignments. We intend to persue methods to alter ProAlign's search algorithm so that it can handle many-to-one alignments. This would involve training from an initial alignment that allows for many-to-one links, such as one of the Candide models. Features that are related to multiple links should be added to our set of feature types, to guide intelligent placement of the many-to-one links.

Unlike the case of many-to-one alignments, where the model already provides some penalty for constraint violations, softening the cohesion constraint would be more difficult. Softening this constraint implies that some penalty for violating phrasal cohesion should be incorporated into the alignment probability model. It is not clear whether or not this could be incorporated as a feature under the current decomposition of the $P(A|E,F)$ model. Extending the model to include cohesion would likely involve incorporating a term that assigns a probability to a given number of overlapping phrases. What this term should be conditioned on, be it word pairs, sentence length or English parse tree structure, is not clear. These issues will hopefully become more clear after more careful study of the nature of genuine cohesion constraint violations and incorrect dependency parses.

6.2 Maximum Entropy

Maximum entropy can be used to improve Candide translation probabilities by using features, such as the improvements to $P(f|e)$ made in [3]. Our $P(A|E,F)$ model already incorporates the same sort of features, modifying $P(l_k|C_k)$ by the features present in the context C_k. However, our model handles features in a Naïve Bayes-like fashion. It assumes conditional independence between features, so that base link probability can be modified by a product of feature ratios. This false assumption can easily create problems, such as links with a context-sensitive probability that is greater than one. This has lead to ProAlign's theoretically unmotivated smoothing method. We hope to use maximum entropy to improve our estimates of $P(l_k|C_k)$.

To incorporate Maximum Entropy, the model derivation would diverge at Equation 3.1:

$$\begin{aligned} P(A|E,F) = P(l_1^t|E,F) &= \prod_{k=1}^{t} P(l_k|E,F,l_1^{k-1}) \\ &= \prod_{k=1}^{t} P(l_k|C_k) \end{aligned}$$

Instead of deriving a feature model at this point, one could replace the $P(l_k|C_k)$ term with a special maximum entropy model trained for all links with the form $\bar{l}_k = l(\overline{e}_{i_k}, \overline{f}_{j_k})$. Let

$P_{\bar{l}_k}(l_k|C_k)$ be this modeled distribution, which models the probability of linking a given type pair under the current context. The new equation for alignment probability would become

$$P(A|E,F) = \prod_{k=1}^{t} P_{\bar{l}_k}(l_k|C_k)$$

and the derivation would end at that point. ProAlign's feature templates could be used easily in this model, once again drawing the features from C_k. During training, the feature and link counts could be sampled in the exact same manner as in ProAlign. The training process would be much slower, but as ProAlign would be switching from a generative model (Naïve Bayes) to a discriminative model (Maximum Entropy), some feature dependencies, especially those involving overlapping features, would be handled in a more reasonable manner. This would hopefully improve performance. Additionally, maximum entropy training has a relatively simple smoothing method [6] in place, which would replace the current m-estimate hybrid that ProAlign currently employs.

6.3 Synchronous Parsing

Currently, ProAlign is unable to implicitly enumerate all of the possible alignments that it can produce. Instead, it samples from alignment space during training, and uses heuristic search to approximate the optimal set of links when aligning text. It has been pointed out by members of the synchronous parsing community that an implicit enumeration of alignment space may be possible under our probabilistic framework. Synchronous parsing, described briefly in Section 2.5, is a dynamic programming method that uses the structure of chart parsing algorithms [25] in order to parse and align bitext simultaneously. Phrasal cohesion and one-to-one alignments are natural side-effects of using synchronous parsing to align text. This implies that if one were to use synchronous parsing as their search method, the space of possible alignments implicitly enumerated by the dynamic programming algorithm could be constrained to one that is identical or similar to that of ProAlign. If this were the case, a version of ProAlign that used synchronous parsing could very well yield similar or better results than the version described in this thesis.

Other attempts to constrain synchronous parsing with an existing dependency tree in order to align text have met with only moderate success [22]. ProAlign, which is constrained in order to maintain phrasal cohesion, has shown itself to be capable of competing with the state of the art in word alignment. We intend to collaborate with Dekai Wu, author of the Stochastic Inversion Transduction Grammar [42] for synchronous parsing, in order to

compare the current version of ProAlign to a version of ProAlign that uses synchronous parsing. Hopefully, a comparison between cohesion constrained search and synchronous parsing will clarify whether or not the two are searching the same alignment spaces, and will reveal which tool is best suited for the word alignment task.

Chapter 7

Conclusion

We have presented ProAlign, a best-first search algorithm designed to iteratively refine an existing word alignment. We have shown that, coupled with the right search constraints, features, and initial alignment, ProAlign is capable of producing high quality word alignments from bitext, without using manually aligned training data. The word alignments produced have been shown to be high quality and very precise. This will allow other systems to make use of the alignments produced by ProAlign without introducing more noise than is necessary into the data stream.

In achieving our goal of creating a high quality word aligner, we have proposed a new statistical model designed specifically for capturing the information present in a noisy word alignment. This model is one of the largest factors in allowing ProAlign to be so effective in refining word alignments. It is our hope that this model is easier to understand than the Candide statistical models that have come before it. Its simple feature model, and the potential opportunity to make use of maximum entropy, allow us to continue to test ideas about word alignment by adding context-related features that may be useful. This has already allowed the discovery of the adjacency and dependency feature templates, which have shown themselves to be effective in informing link placement.

The experiments conducted here have demonstrated the validity of ProAlign as a complete alignment algorithm. It is competitive with the current state of the art alignment method, and it has shown itself to be one of the top two methods at a recent international alignment algorithm comparison. It has been tested with multiple language pairs, and with drastically different sizes of parallel corpora. Furthermore, ProAlign's use of features and constraints has been evaluated, and shown to be useful. Finally, the $P(A|E,F)$ decomposition presented here has been shown to be better-suited to word alignment refinement than the equivalent decomposition provided by Candide Model 1.

Bibliography

[1] Y. Al-Onaizan, J. Curin, M. Jahr, K. Knight, J. Lafferty, I. D. Melamed, F. J. Och, D. Purdy, N. A. Smith, and D. Yarowsky. Statistical machine translation: Final report. In *Summer Workshop on Language Engineering*, John Hopkins University Center for Language and Speech Processing, 1999.

[2] H. Alshawi and S. Douglas. Learning dependency transduction models from unannotated examples. *Philosophical Transactions of the Royal Society (Series A: Mathematical, Physical and Engineering Sciences)*, 358:1357–1372, 2000.

[3] A. L. Berger., S. A. Della Pietra, and V. J. Della Pietra. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22(1):39–71, 1996.

[4] P. F. Brown, S. A. Della Pietra, V. J. Della Pietra, and R. L. Mercer. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–312, 1993.

[5] J. Carbonell, K. Probst, E. Peterson, C. Monson, A. Lavie, R. Brown, and L. Levin. Automatic rule learning for resource-limited mt. In *Proceedings of AMTA-02*, pages 1–10, 2002.

[6] S. F. Chen and R. Rosenfeld. A survey of smoothing techniques for maximum entropy models. *IEEE Transactions on Speech and Audio Processing*, 8(1):37–50, 2000.

[7] C. Cherry and D. Lin. A probability model to improve word alignment. In *Meeting of the Association for Computational Linguistics*, Sapporo, Japan, July 2003.

[8] J. N. Darroch and D. Ratcliff. Generalized iterative scaling for log-linear models. *Annals of Mathematical Statistics*, 43(5):1470–1480, October 1972.

[9] H. Dejean, E. Gaussier, C. Goutte, and K. Yamada. Reducing parameter space for word alignment. In Rada Mihalcea and Ted Pedersen, editors, *HLT-NAACL 2003 Workshop: Building and Using Parallel Texts: Data Driven Machine Translation and Beyond*, pages 23–26, Edmonton, Alberta, Canada, May 31 2003. Association for Computational Linguistics.

[10] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, 39(1):1–38, 1977.

[11] T. E. Dunning. Accurate methods for the statistics of surprise and coincidence. *Computational Linguistics*, 19(1):61–74, 1993.

[12] H. J. Fox. Phrasal cohesion and statistical machine translation. In *Proceedings of EMNLP-02*, pages 304–311, 2002.

[13] W. Gale, K. Church, and D. Yarowsky. A method for disambiguating word senses in a large corpus. *Computers and Humanities*, 26:415–439, 1993.

[14] W. A. Gale and K. W. Church. Identifying word correspondences in parallel texts. In *4th Speech and Natural Language Workshop*, pages 152–157. DARPA, Morgan Kaufmann, 1991.

[15] W. A. Gale and K. W. Church. A program for aligning sentences in bilingual corpora. In *Meeting of the Association for Computational Linguistics*, pages 177–184, 1991.

[16] K. Knight. Automating knowledge acqusition for machine translation. *AI Magazine*, 18(4), 1997.

[17] K. Knight. A statistical MT tutorial workbook. Unpublished, available at http://www.isi.edu/ knight, August 1999.

[18] P. Koehn, F. J. Och, and D. Marcu. Statistical phrase-based translation. In Marti Hearst and Mari Ostendorf, editors, *HLT-NAACL 2003: Main Proceedings*, pages 127–133, Edmonton, Alberta, Canada, May 27 - June 1 2003. Association for Computational Linguistics.

[19] D. Lin and C. Cherry. Linguistic heuristics in word alignment. In *Meeting of the Pacific Association for Computational Linguistics*, Halifax, Nova Scotia, Canada, August 2003. To appear.

[20] D. Lin and C. Cherry. Proalign: Shared task system description. In Rada Mihalcea and Ted Pedersen, editors, *HLT-NAACL 2003 Workshop: Building and Using Parallel Texts: Data Driven Machine Translation and Beyond*, pages 11–14, Edmonton, Alberta, Canada, May 31 2003. Association for Computational Linguistics.

[21] D. Lin and C. Cherry. Word alignment with cohesion constraint. In Marti Hearst and Mari Ostendorf, editors, *HLT-NAACL 2003: Short Papers*, pages 49–51, Edmonton, Alberta, Canada, May 27 - June 1 2003. Association for Computational Linguistics.

[22] A. Lopez, M. Nossal, R. Hwa, and P. Resnik. Word-level alignment for multilingual resource acquisition. In *Proceedings of the Workshop on Linguistic Knowledge Acquisition and Representation: Bootstrapping Annotated Language Data*, 2002.

[23] E. Macklovitch. Transcheck - or the automatic validation of human translations. In *Proc. of the MT Summit V*, Luxembourg, 1995.

[24] R. Malouf. A comparison of algorithms for maximum entropy parameter estimation. In *Proc. of the Sixth Conference on Natural Language Learning (CoNLL-2002)*, pages 49–55, 2002.

[25] C. D. Manning and H. Schütze. *Foundations of Statistical Natural Language Processing*. The MIT Press, 2001.

[26] D. Marcu. Towards a unified approach to memory- and statistical-based machine translation. In *Proc. of the Meeting of the Association for Computational Linguistics*, Toulouse, France, July 2001.

[27] J. Martin, H. Johnson, B. Farley, and A. Maclachlan. Aligning and using an English-Inuktitut parallel corpus. In Rada Mihalcea and Ted Pedersen, editors, *HLT-NAACL 2003 Workshop: Building and Using Parallel Texts: Data Driven Machine Translation and Beyond*, pages 115–118, Edmonton, Alberta, Canada, May 31 2003. Association for Computational Linguistics.

[28] I. D. Melamed. Automatic construction of clean broad-coverage translation lexicons. In *2nd Conference of the Association for Machine Translation in the Americas*, pages 125–134, Montreal, 1996.

[29] I. D. Melamed. Automatic discovery of non-compositional compounds in parallel data. In Claire Cardie and Ralph Weischedel, editors, *Proceedings of the Second Conference on Empirical Methods in Natural Language Processing*, pages 97–108. Association for Computational Linguistics, Somerset, New Jersey, 1997.

[30] I. D. Melamed. A portable algorithm for mapping bitext correspondence. In P. R. Cohen and W. Wahlster, editors, *Meeting of the Association for Computational Linguistics*, pages 305–312, Somerset, New Jersey, 1997.

[31] I. D. Melamed. Models of translational equivalence among words. *Computational Linguistics*, 26(2):221–249, 2000.

[32] I. D. Melamed. Multitext grammars and synchronous parsers. In Marti Hearst and Mari Ostendorf, editors, *HLT-NAACL 2003: Main Proceedings*, pages 158–165, Edmonton, Alberta, Canada, May 27 - June 1 2003. Association for Computational Linguistics.

[33] R. Mihalcea and T. Pedersen. An evaluation exercise for word alignment. In Rada Mihalcea and Ted Pedersen, editors, *HLT-NAACL 2003 Workshop: Building and Using Parallel Texts: Data Driven Machine Translation and Beyond*, pages 1–10, Edmonton, Alberta, Canada, May 31 2003. Association for Computational Linguistics.

[34] T. Mitchell. *Machine Learning*. WCB/McGraw-Hill, 1997.

[35] K. Nigam, J. Lafferty, and A. McCallum. Using maximum entropy for text classification. In *IJCAI-99 Workshop on Machine Learning for Information Filtering*, pages 61–67, 1999.

[36] F. J. Och and H. Ney. A comparison of alignment models for statistical machine translation. In *COLING'00: Proc. of the 18th Int. Conf. on Computational Linguistics*, pages 1086–1090, Saarbrücken, Germany, August 2000.

[37] F. J. Och and H. Ney. Improved statistical alignment models. In *Meeting of the Association for Computational Linguistics*, pages 440–447, Hong Kong, China, October 2000.

[38] S. A. Della Pietra, V. J. Della Pietra, and J. D. Lafferty. Inducing features of random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(4):380–393, 1997.

[39] L. Rabiner and B. H. Juang. *Fundamentals of Speech Recognition*, pages 321–389. Prentice Hall, Upper Saddle River, NJ, 1982.

[40] J. Véroins. From the Rosetta Stone to the information society: A survery of prallel text processing. In J. Véroins, editor, *Parallel text processing: Alignment and use of translation corpora*, pages 1–24. Kluwer Academic Publishers, 2000.

[41] S. Vogel, H. Ney, and C. Tillmann. HMM-based word alignment in statistical translation. In *Proc. of the International Conference on Computational Linguistics*, pages 836–841, Copenhagen, Denmark, August 1996.

[42] D. Wu. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational Linguistics*, 23(3):374–403, 1997.

[43] K. Yamada and K. Knight. A syntax-based statistical translation model. In *Meeting of the Association for Computational Linguistics*, pages 523–530, 2001.